NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

POST DEPLOYMENT SOFTWARE SUPPORT OF THE U.S. ARMY'S SPECIAL OPERATIONS AIRCRAFT: A SOFTWARE ACQUISITION MANAGEMENT CASE STUDY

by

Scott C. Dolloff

December 1995

Principal Advisor: **Martin J. McCaffrey**

Approved for public release; distribution is unlimited

19960405 069

DTIC QUALITY INSPECTED 1

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
December 1995

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE
POST DEPLOYMENT SOFTWARE SUPPORT OF THE U.S. ARMY'S SPECIAL OPERATIONS AIRCRAFT: A SOFTWARE ACQUISITION MANAGEMENT CASE STUDY

5. FUNDING NUMBERS

6. AUTHOR(S)
Dolloff, Scott, C.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This thesis examines the issues faced by the Program Manager in providing for Post Deployment Software Support (PDSS) of the U.S. Army's Special Operations Aircraft, MH-60K and MH-47E. PDSS of Department of Defense weapon systems is becoming increasingly important for several reasons. First, weapon systems functions are migrating from hardware to software. Second, these functions are migrating to software because it is flexible. Third, because software is flexible, it continues to evolve throughout its lifecycle. Finally, this evolution accounts for approximately 70 percent of the lifecycle cost of software. This thesis presents a case study of PDSS of the U.S. Army's Special Operations Aircraft. The case analysis identifies significant management issues in the following areas: acquisition management, project management, and software engineering. Conclusions drawn from the analysis reveal that the success of PDSS is dependent on effective program management. Effective management of a software acquisition involving PDSS requires the following: planning; a cooperative buyer-seller relationship; selection of the proper contract type; technical support resources; and the use of proven software management techniques such as metrics and process maturity assessments. Implementing the recommendations included in this thesis should improve the future management of PDSS.

14. SUBJECT TERMS
Software Acquisition Management, Software Management, Software Maintenance, Software Support, PDSS

15. NUMBER OF PAGES
155

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

Approved for public release; distribution is unlimited

POST DEPLOYMENT SOFTWARE SUPPORT OF THE U.S. ARMY'S SPECIAL OPERATIONS AIRCRAFT: A SOFTWARE ACQUISITION MANAGEMENT CASE STUDY

Scott C. Dolloff
Major, United States Army
B.S., Bridgewater State College, 1983

Submitted in partial fulfillment of the
requirement for the degree of

MASTER OF SCIENCE IN MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
December 1995

Author: Scott C Dolloff
Scott C. Dolloff

Approved by: Martin J. McCaffrey
Martin J. McCaffrey, Principal Advisor

W. Max Woods
W. Max Woods, Associate Advisor

Reuben T. Harris
Reuben T. Harris, Chairman
Department of Systems Management

ABSTRACT

This thesis examines the issues faced by the Program Manager in providing for Post Deployment Software Support (PDSS) of the U.S. Army's Special Operations Aircraft, MH-60K and MH-47E. PDSS of Department of Defense weapon systems is becoming increasingly important for several reasons. First, weapon systems functions are migrating from hardware to software. Second, these functions are migrating to software because it is flexible. Third, because software is flexible, it continues to evolve throughout its lifecycle. Finally, this evolution accounts for approximately 70 percent of the lifecycle cost of software. This thesis presents a case study of PDSS of the U.S. Army's Special Operations Aircraft. The case analysis identifies significant management issues in the following areas: acquisition management, project management, and software engineering. Conclusions drawn from the analysis reveal that the success of PDSS is dependent on effective program management. Effective management of a software acquisition involving PDSS requires the following: planning; a cooperative buyer-seller relationship; selection of the proper contract type; technical support resources; and the use of proven software management techniques such as metrics and process maturity assessments. Implementing the recommendations included in this thesis should improve the future management of PDSS.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank several people who made this thesis possible. First, I am most grateful to my family - Lori, Allision, and Jonathan - for their patience, understanding, and encouragement. Second, to Mr. Randy Buckner and Major Jerry Hopkins who took me on as part of the team in researching the SOA PDSS. My primary advisor - Professor Martin J. McCaffrey who encouraged my learning about the challenges of software management through the thesis process, and who reviewed and improved the final product. Finally, my associate advisor - Professor W. Max Woods who graciously agreed to advise me and to review my work.

I. INTRODUCTION

A. GENERAL

Over the past 30 years, the Department of Defense (DoD) has embraced computer technology along with the private sector. As a result, increasingly complex weapon systems have been produced. Systems such as the Vietnam era F-4 aircraft, which had virtually no computer code, have been replaced over time by systems such as the F-16D which has approximately 236,000 lines of computer code. [Ref. 1, p. 30] While software may offer the promise of increased system flexibility and capability, and reduced operator workload, the software development process that yields these benefits is still inherently risky.

DoD's most recent experience with the complexities and risks in the acquisition of Mission Critical Computer Resources (MCCR), weapon system computers and their software, has been illustrated in systems such as the Air Force's C-17 cargo aircraft, the Navy's F-14D fighter aircraft, and the Army's Fire Direction Data Manager. These and numerous other systems have been plagued with problems such as long development schedules, ill-defined requirements, the inability to meet performance specifications, and cost overruns. From its investigation of DoD's software troubles, the General Accounting Office (GAO) concluded in a December 1992 report [Ref. 2], that DoD's software problems fall into three general categories: management; requirements definition; and testing.

Why does DoD have these software problems? The Defense Systems Management College's Mission Critical Computer Resources Management Guide [Ref. 3, pp. 2-6 - 2-8] provides some insight into the causes of DoD's software problems. Among these are:

- Weapon system software has extremely demanding requirements which drive system designers toward extremely complex solutions.
- The requirements for the software are usually not defined until late in the development cycle, and then often continue to expand.

- Weapon system software with its real-time and fault free operating requirements is the most difficult type of software to develop.
- Despite the complex demands placed on software, development budgets and schedules are overly optimistic.
- There is a lack of experienced software managers within the Government.
- The supply of skilled software developers cannot meet the demands of both the military and the civilian market.
- Software engineering is still an immature discipline, and is just now beginning to develop and apply structured methods in software development.

Despite these problems, weapon systems software has become a system unto itself. It is essentially the "glue" [Ref. 1, p. 32] that cements together and integrates the various sub-systems of the weapon system. Because of its central role in modern weapon system design, software is now the critical path in system development and the key to meeting demanding performance requirements.

As weapon system software has grown in importance, so have the costs and challenges associated with maintaining it. In 1992, the GAO estimated that DoD spends from $24 billion to $32 billion on software annually. This accounts for between eight and 11 percent of the Defense budget. [Ref. 4] Over the system life-cycle, as much as 70 percent [Ref. 3, p. 7-3] of the cost of software systems is spent on support and maintenance. As current and future weapon systems mature, and as the inventory of weapon systems software continues to grow, the area of software maintenance (what DoD calls Post Deployment Software Support (PDSS)) will become increasingly expensive. One of the primary causes of the high cost of software maintenance is that unlike hardware, software tends to decay under the impact of the many small changes that users request in an effort to enhance a system's functionality. [Ref. 5, p. 147]

In fact the term "software maintenance" is somewhat misleading. It generally includes not only the fixing of software defects, but also includes enhancements which meet new user requirements, and improvements in the software's quality. While many weapon systems have their PDSS performed by an organic DoD software support activity, some systems depend on the software development contractor for support. One such system is the U.S. Army's Special Operations Aircraft (SOA).

The SOA program was an Acquisition Category (ACAT) II program. It originally began as a Non-developmental Item (NDI) acquisition to provide 26 MH-47E and 23 MH-60K aircraft for the U.S. Amy Special Operations Command. The SOA program modified the Boeing CH-47D and the Sikorsky UH-60L helicopter airframes to provide Special Operating Forces (SOF) aviation with an enhanced capability to conduct clandestine, deep penetration airlift missions, under adverse weather conditions, over all types of terrain. [Ref. 6, pp. 2-8 - 2-12] The heart of the system, which is identical on both airframes, is the software intensive Integrated Avionics Subsystem (IAS). This system, developed by IBM Federal Systems Division (now LORAL Federal Systems) under subcontracts with the aircraft manufacturers Boeing and Sikorsky, integrates a total software system which consists of approximately 380,000 source lines of computer code.

The SOA Program Computer Resources Management Plan [Ref. 7] originally specified that the PDSS of the fielded IAS would be performed by the Software Engineering Directorate at the U.S. Army Communications-Electronics Command (CECOM). However, a 1993 Process Action Team (PAT), commissioned by the Product Manager, conducted a cost-benefit analysis [Ref. 8] and recommended that PDSS of the Special Operations Aircraft should be performed by the IAS manufacturer, LORAL.

Since that time, the aircraft have been fielded and program management responsibility has transitioned from the Product Manager Special Operations Aircraft (PM SOA) to the Technology Applications Program Office (TAPO). This office is subordinate to the U.S. Army Special Operations Command, and co-located with the Army Aviation and Troop Command (ATCOM) in St. Louis. Since July 1994, the PDSS of the IAS has been performed by LORAL under an acquisition strategy which includes a one year firm-fixed-price contract with two one year options. The Program Manager (PM) is dissatisfied with this PDSS arrangement for a number of reasons, chief being the first year cost of $4.45 million, which he feels is too much. Nonetheless, he has elected to keep the PDSS effort with LORAL for at least the near term while he learns more about management of software acquisition and software support and develops a long range acquisition strategy for PDSS.

B. AREA OF RESEARCH

Post Deployment Software Support is but one of the many issues involved in DoD software acquisition management. This thesis will examine the issues faced by the Program Manager of the Special Operations Aircraft for PDSS. The primary objective of the thesis is to identify and analyze the significant issues associated with the current PDSS program and to recommend actions to address these issues. The secondary objective of the thesis is to document the lessons learned from the SOA program's PDSS experience to assist other DoD Program Managers in planning and conducting PDSS.

C. RESEARCH QUESTIONS

The research questions for this thesis are divided into two categories: primary and subsidiary.

1. Primary

In light of the Program Manager's PDSS situation, the primary research question for this thesis is: What are the significant management issues associated with the current SOA Post Deployment Software Support program and what actions can be taken to address these issues?

2. Subsidiary

The following subsidiary questions support this research:

- What is PDSS and what are its essential elements?
- What are the principal policies and guidance concerning PDSS?
- What are the elements of the current SOA PDSS program?

- What are the significant management issues associated with the current PDSS program?
- What actions can be taken to address these issues?
- What lessons can be learned from the SOA program's experience with PDSS?

D. SCOPE

This thesis will be limited to a discussion of the software acquisition management issues facing the Program Manager in providing Post Deployment Software Support for the Special Operations Aircraft. While there are technical issues involving software engineering processes, these will be addressed only to the degree necessary to clarify the management situation.

E. METHODOLOGY

Analysis of the research questions was accomplished using three techniques. First a literature search was conducted for both software maintenance and support, and the SOA Program. Information was obtained from: the Defense Logistics Studies Information Exchange (DLSIE); the Defense Technological Information Center (DTIC); the Air Force's Software Technology Support Center; the Institute of Electrical and Electronics Engineers (IEEE) data base and other sources from the Dudley Knox Library at the Naval Postgraduate School; PM SOA; TAPO; and the Army's Office of the Director Information Systems for Command, Control, Communications, and Computers (ODISC4). Second, interviews were conducted with personnel from PM SOA, TAPO, ATCOM, and the PDSS contractor, LORAL. Finally, the analysis of the issues identified with the current PDSS program was conducted in light of the literature review, interviews, and the researcher's knowledge and insight gained from personal involvement with the SOA program. The result of this analysis is a set of recommendations to address the PDSS program issues.

F. ORGANIZATION

Chapter II presents the background. It addresses the history of DoD's experience with weapon systems software acquisition, explains players and the processes involved in the DoD software acquisition environment, examines key elements of software support, and reviews the principal policies and the guidance concerning PDSS.

Chapter III provides the relevant history of the Special Operations Aircraft Program, and documents the case description of the current PDSS program. It summarizes the key program decisions, provides a description of the aircraft and software systems, and explains the key elements of the current PDSS acquisition management process.

Chapter IV presents the analysis of the issues associated with the current PDSS program. This analysis will focus on the acquisition management, project management, and software engineering issues.

Chapter V presents the conclusions and the recommendations to address the issues identified and analyzed in Chapter IV. It will also provide the answers to the research questions and areas for further research.

II. BACKGROUND

A. INTRODUCTION

This chapter will provide the background of the DoD software acquisition management environment and PDSS. The chapter is divided into three sections. The first section provides a history of DoD's experience in acquiring weapon systems software. The second section will address the DoD software acquisition management environment, including the players and their processes, and the policies which guide them. Finally, the third section will address software support and PDSS.

B. SOFTWARE ACQUISITION - THE HISTORICAL CONTEXT

1. Software Engineering is the Foundation

The Defense Systems Management College's Mission Critical Computer Resources (MCCR) Guide [Ref. 3] points out that the development of computer software as a recognized activity is less than 40 years old. The nation's universities did not establish separate computer science departments until the late 1960s, and the term "software engineering" did not appear until 1968. Therefore, unlike other engineering disciplines, software engineers lack any significant body of time-tested practices, procedures, and tools normally available in other engineering disciplines. [Ref. 3, p. 2-1] The MCCR Guide quotes author Richard Fairley's definition of software engineering as:

> ...the technological and managerial discipline concerned with systematic production and maintenance of software products that are developed and modified on time and within cost estimates. [Ref. 3, p. 5-1]

It is important to understand that software engineering, the foundation of the process by which software is developed, must be scientific and disciplined. Until very recently, it has not. While the trend is slowly changing, software engineering is relatively still in its infancy, a rather immature discipline involving too much art and craft, and not enough structure - an activity which has been called a "black art." [Ref. 1, p. 30]

2. Software's Growing Role

In spite of the fact that software engineering is immature, DoD's appetite for complex weapon systems computers and software has grown dramatically over time. In his article, *Is Software DOD's Achilles' Heel?*, [Ref. 1] James Kitfield examines the possibility that DoD's demand for high quality weapon systems software may someday exceed the nation's capacity to supply it. Kitfield points out that in the software intensive aircraft avionics market alone, demand for software increases 25 percent annually. In the same period, the education system produces only four percent more software programmers. He provides insight into the tremendous leaps made in weapon systems software content since the Vietnam War-era with the figures shown in Table 1.

SYSTEM	LINES OF SOURCE CODE
F-4	Virtually None
F-16D	236,000
C-17	625-000 to 750,000
B-1B	1,200,000
ATF	5,000,000 to 7,000,000
SDI	25,000,000

Table 1. Weapon Systems Software Growth

As weapon systems have increased in software content, designers have placed increasing requirements on software to perform functions which had been previously performed by hardware. In performing these functions, software offers both increased system flexibility and reduced operator workload. As shown in Figure 1., the trend calls for an ever increasing role for weapon systems software.

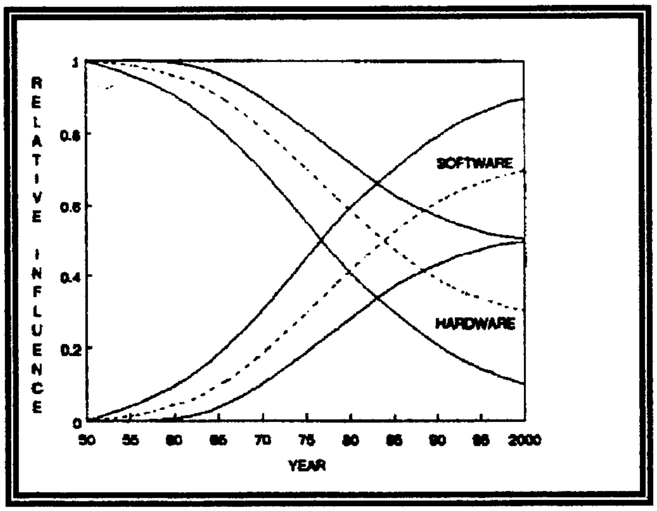

Figure 1. Software's Growing Influence
From [Ref. 3, p. 2-3]

Given its increasing role, software has become a system unto itself, integrating all the various functions and sub-systems of the weapon system. But this role is not without a degree of risk. Despite the difficulties in engineering complex software, system's designers frequently depend heavily on it to solve performance problems. In this regard, Kitfield quotes Jack Kramer, director of software engineering for the Institute for Defense Analysis:

> Because software is the glue that cements the weapon together, all the problems that they (*the designers*) were unable to solve elsewhere in the system - which are obviously some of the hardest - are pushed onto software.

Kramer goes on to warn that what may seem to a DoD program manager to be a trivial software change, may have dramatic consequences. "Program managers wouldn't think of adding a second engine to an aircraft at the last minute, but they sometimes try similar changes in software."

3. Software Acquisition Problems and Risks

Along with software's increasing role have come numerous problems and risks. The subject of numerous General Accounting Office (GAO) reports, these problems, and their underlying causes were summarized in a 1992 report. [Ref. 2] While the program specific problems differed, the GAO found that they have occurred in three general areas as outlined below:

- **Management**
  - Lack of management attention/oversight.
  - Lack of adequate software management concepts, methods, practices.
  - Lack of adequate planning.
  - Development proceeded despite serious problems.
- **Requirements Definition**
  - Lack of well-defined requirements.
  - Requirements change to meet new missions.
  - Lack of overall system perspective.
  - System not ready to adapt to change.
  - Software products cannot/may not meet security requirements.
- **Testing**
  - Lack of adequate testing methods and approaches.
  - Lack of system-level integration testing.

The final impact of these problems has usually been a reduced level of system performance or the outright inability of the system to meet its requirements. Along with these deficiencies also come inevitable cost and schedule overruns. But why have DoD software development programs experienced these problems? The same GAO report highlights some of the reasons behind these software acquisition difficulties:

- Mission critical systems require millions of lines of highly complex software.
- The process to develop high quality software is poorly understood.
- Unlike hardware, it is difficult to measure software's essential characteristics, such as correctness, robustness, performance, and integrity.
- Developers have difficulty in measuring the progress of development.
- These systems must operate in a real-time mission environment.

Given these problems, it is clear that software development has historically been inherently risky and difficult for the DoD Program Manager. But these software development problems and risks are not unique to DoD. From his company's study of several thousand software projects, both military and civilian, Software Productivity Research's founder Capers Jones lists what he has found to be the 10 most serious software risks: [Ref. 5, p. 47-60]

- **Inaccurate Metrics** - Inaccurate measures of productivity such as the commonly used "Lines of Code" (LOC) or "Source Lines of Code" (SLOC).
- **Inadequate Measurement** - Project cost tracking and cost collection systems omit or "leak" major portions of project cost.
- **Excessive Schedule Pressure** - A software development schedule set by some arbitrary method rather than by a realistic estimate of the effort required to complete the project.
- **Management Malpractice** - Managers who are not properly trained or skilled to meet the challenges of software development.
- **Inaccurate Cost Estimating** - Under estimating the cost of the software project.
- **Silver Bullet Syndrome** - Management's belief that a single tool or method will solve all of their software problems.
- **Creeping User Requirements** - A constant flow of new and unanticipated requirements changes during software development.
- **Low Quality** - Significant user dissatisfaction with the delivered product or high software defect rates.
- **Low Productivity** - Projects which fail to proceed at the rate which management expects.
- **Canceled Projects** - The "average" canceled project is about a year late and approaches or exceeds twice its planned budget.

The GAO's findings [Ref. 2] certainly indicate that DoD shares many of these common software risks with the private sector. However, Jones also provides what he has found to be the most common risks in DoD specific software projects: [Ref. 5, p. 33-35]

- **Excessive Paperwork** - This risk is present in 90% of all DoD projects. The paperwork or documentation traditionally required by Military Standards such as

DoD-STD-2167A often accounts for as much as 52% of the total cost of the software project.

- **Low Productivity** - Present in 85% of DoD Projects. Jones defines this risk as projects which are 50% lower in productivity than the U.S. average.
- **Long Schedules** - Present in 75% of projects. While the term "long schedule" is relative, Jones believes that military projects which last more than five years will have troubles adapting to a world which is changing faster than the software is developing.
- **Creeping User Requirements** - Present in 70% of projects. Jones has found that on a typical four to five year project, as much as 50 to 60% of the delivered software functionality is in the form of creeping requirements.
- **Unused or Unusable Software** - Present in 45% of projects. This risk is tied to the long development schedules and the fact that the computer hardware on which the software was built to run is outdated by the time the project is completed.

4. Management is the Key

The problems and risks associated with DoD software development projects have given rise to the perception that there is in DoD, and in the software industry in general, a "software crisis." But author Robert L. Glass responds to this perception by asking a rather probing question: "If there is truly a software crisis, then how could we be sending vehicles into space?" He goes on to theorize that the root cause of the "crisis" is not in how we build software, but rather in how we *think* we build software. He believes that the problem lies not in the state of software engineering technology, but rather in how we ***manage*** that technology. [Ref. 9, p. 27]

This observation concerning management is echoed both by the GAO [Ref. 2], and the Defense Science Board Task Force on Military Software, [Ref. 10, p. 7] which stated in its 1987 report that:

> In spite of substantial technical development needed in requirements setting, metrics and measures, tools, etc., the Task Force is convinced that today's major problems with military software development are not technical problems, but managerial problems.

While much of the criticism in the report is directed toward corporate-level DoD software management, the ultimate responsibility for the success or failure of a weapon system acquisition involving software lies with the Program Manager. Management at the project level is key to successful software development. The next section describes the environment in which the Program Manager of a software-intensive system must operate.

C. THE SOFTWARE ACQUISITION MANAGEMENT ENVIRONMENT

This section describes the software acquisition management environment. This environment is described in terms of its key players, and the processes they use in developing weapon systems software.

1. The Players and Processes

Marciniak and Reifer [Ref. 11, p. 50] define software acquisition management as, "...the process of acquiring custom software, usually by contract from some third party." It includes activities such as planning, contracting, budgeting, evaluating and managing performance, and providing for the future support of the system. As shown in Figure 2, the primary players involved in the software acquisition management process are the customer or user, the contracting activity or buyer, and the developer or seller.

Within the context of the DoD acquisition environment, the user is the activity which has generated the Mission Need Statement and the Operational Requirements Document for the system, this activity is often referred to as the Combat Developer. The buyer is the DoD Program Manager, or the Materiel Developer. Finally, the developer is the contractor, or in some cases an organic DoD software development activity. Marciniak and Reifer point out that the process of acquiring weapon system software requires the interaction of these players in three overlapping processes as shown in Figure 3. The acquisition management process is primarily the responsibility of the Program Manager, the software engineering process belongs to the contractor, while both parties share responsibility for project management.

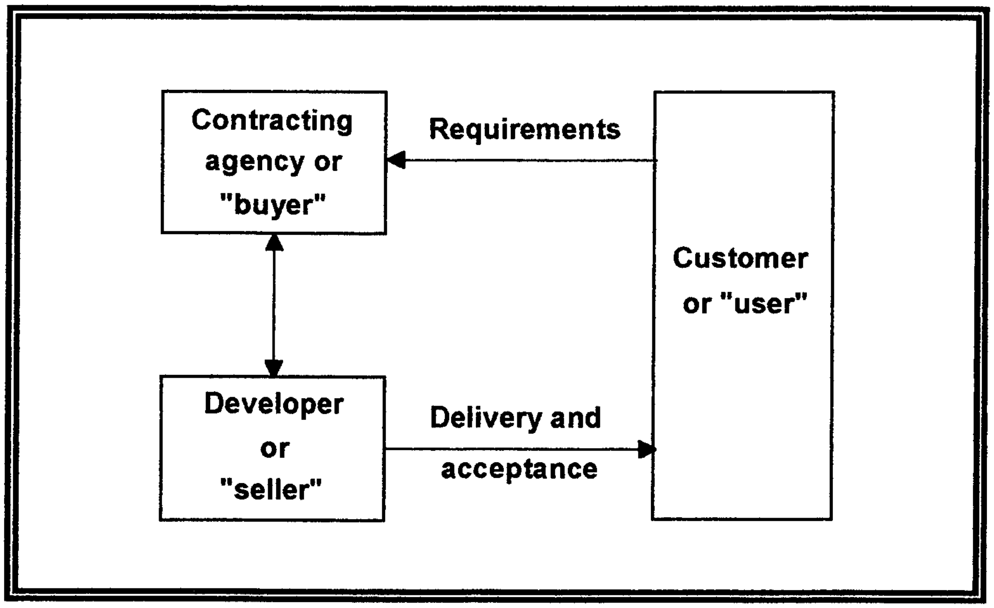

Figure 2. The Software Acquisition Management Players From [Ref. 11, p. 51]

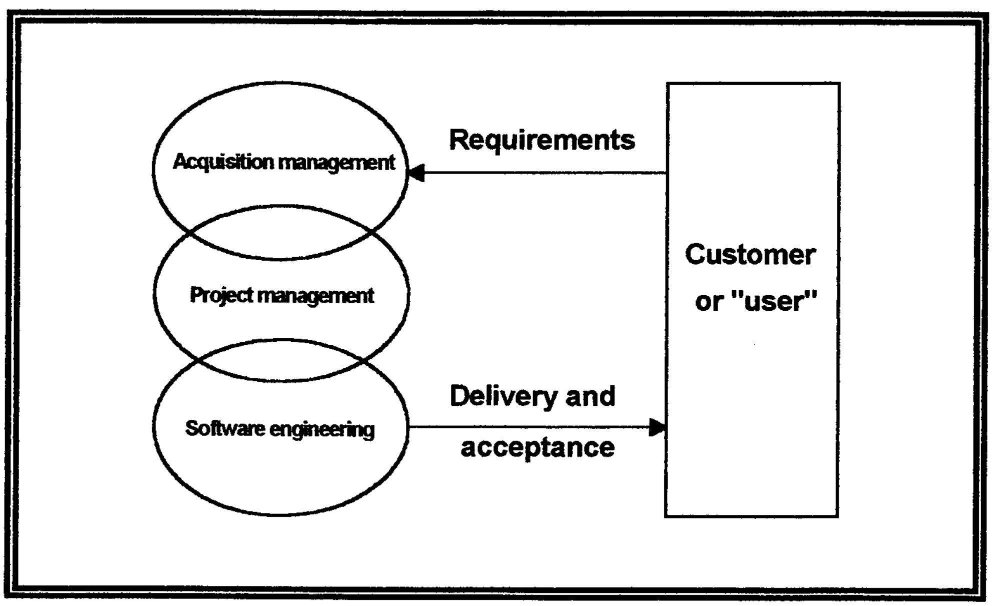

Figure 3. The Software Acquisition Management Relationships From [Ref. 11, p. 51]

2. Acquisition Management

The DoD software acquisition management process is influenced by both policies and standards. This section discusses the key policies and standards which impact software acquisition.

a. Acquisition Management Policy

DoD weapon systems acquisition, including the software component of the system is conducted within the policy framework established by DoD Directive 5000.1, Defense Acquisition, [Ref. 12] and DoD Instruction 5000.2, Defense Acquisition Management Policies and Procedures [Ref. 13]. Together, these policies establish a disciplined system acquisition life-cycle of discrete phases which include: (1) Phase 0 - Concept Exploration and Definition; (2) Phase I - Demonstration and Validation; (3) Phase II- Engineering and Manufacturing Development; (4) Phase III - Production and Deployment; (5) Phase IV - Operations and Support. At the beginning of each phase there is a milestone decision point at which the status of the program is reviewed to determine if it should proceed to the next phase. The ultimate objective of this management framework is to "translate operational needs into stable, affordable programs." [Ref. 12, p. 1-1] The major activities within this process are: [Ref. 12, pp. 1-4 - 1-6]

- Development of acquisition strategies and program plans.
- Risk Management.
- Contract type selection.
- Development of program objectives and baselines.
- Competition and source selection.
- Contractor management.

b. Software Standards

In addition to DoD Directive 5000.1 and DoD Instruction 5000.2 which provide the high-level acquisition guidance, the DoD acquisition environment has also been heavily influenced by specifications and standards. The extent of this influence has been reduced by Defense Secretary William Perry's June 1994 memorandum, Specifications and Standards - A New Way of Doing Business. [Ref. 14] Because the software industry has been slow to develop its own military related standards, the following standards have had a great impact, and will continue to impact software-intensive programs for the near future:

- **DoD-STD-2167A** - Defense System Software Development. [Ref. 15] This was DoD's previous attempt to bring discipline to the weapon systems software development process. It detailed the requirements for software development, testing, configuration management, documentation, and transition of support. It has been superseded by MIL-STD-498, Software Development and Documentation and is used in ongoing programs only.
- **DoD-STD-2168** - Defense System Software Quality Program. [Ref. 16] Established quality control of the software development process and products.
- **MIL-STD-498** - Software Development and Documentation. [Ref. 17] Provides a single standard for all DoD software development, both weapon systems and information systems. It is interesting to note that MIL-STD-498 was approved after the Defense Secretary's policy on minimizing the use of such standards.

3. Project Management

Project management bridges the gap between the acquisition management and software engineering processes. Because it is primarily concerned with control of the software engineering process, the primary player in this process is the software development contractor. However, project management intersects acquisition management at the project level, the realm of the DoD Program Manager, and is one of his primary responsibilities.

a. Definition

Software project management is defined as:

> ...a system of procedures, practices, technologies, and know-how that provides the planning, organizing, staffing, directing, and controlling necessary to successfully manage an engineering project. [Ref. 18, p. 15]

b. Project Management Activities

Some of the key management activities which occur within each of the functions included in the definition are: [Ref. 18, p. 17]

- **Planning**
  - Set objectives or goals.
  - Develop strategies.
  - Determine courses of action and make decisions.
  - Prepare budgets.
  - Document project plans.
- **Organizing**
  - Select and establish organizational structures.
  - Define responsibilities.
  - Establish position qualifications.
  - Document organizational structures.
- **Staffing**
  - Fill organizational positions.
  - Educate or train personnel.
  - Provide for general development.
  - Document staffing decisions.
- **Directing**
  - Provide leadership.
  - Supervise personnel.
  - Coordinate activities.
  - Facilitate communications.
  - Manage changes.

- **Controlling**
  - Develop standards of performance.
  - Establish monitoring and reporting systems.
  - Measure results.
  - Initiate corrective actions.
  - Document controlling methods.

Each of these topics is addressed in the Program Manager's project management plan. [Ref. 19, p. 124] It should be prepared early in the project and be included as part of the solicitation package that is provided to the contractor. The Program Manager's plan will provide the basis for the contractor's management plan which will be documented in a similar project management plan or in the software development plan.

c. Management Indicators

Of particular concern to DoD Program Managers in software acquisition, is how they will manage and assess contractor's performance during contract execution. Marciniak and Reifer believe that:

> The key to performance management is providing the buyer and seller with visibility into the processes and products of software development. Without visibility, both parties are blind. Neither has insight into whether real progress is being made. To gain visibility, meaningful management reviews need to be held. Configuration management and quality assurance systems need to be put into place, along with a progressive test and evaluation program. Metrics need to be captured and reported to provide management with "hard" evidence or indicators of progress. [Ref. 11, p. 71]

As stated above, in addition to a system of management reviews, and sound processes for configuration management, quality, and testing, one way in which to provide the Program Manager with insight into the contractor's performance is the use of management indicators or metrics. There are numerous software metrics which can be collected. However, the use of metrics requires data collection and therefore adds to the cost of the effort. Program Managers should, to the maximum extent possible, work within the limits of a contractor's existing processes and define a set of metrics which will increase insight into the software development process and pinpoint problem areas. The following is a list of possible management indicators recommended by Marciniak and Reifer: [Ref. 19, pp. 139-149]

- **Memory Utilization** - Measures the utilization of computer memory across the project.
- **Problem Reports** - Problems are anomalies detected during the software development process. This metric measures the open problem report trend and the number of problems relative to the total source lines of code.
- **Software Size** - This metric provides a simple way of tracking development progress by lines of code.
- **Personnel Stability** - This indicator tracks the total number of people assigned to the project by the contractor.
- **Development Progress** - This indicator tracks progress toward key activities in the software development process.
- **Incremental Releases** - Releases are versions of the software produced at various times throughout the development. Each release is produced for a specific purpose and incorporates corrections and added capabilities developed since the last release. This indicator measures the contents of each release.
- **Test Progress** - This metric measures the testing progress based on the test scheduled and tests successfully conducted.
- **Documentation** - This indicator tracks the progress of documentation development against the schedule.

4. Software Engineering

Software engineering is the process by which the software contractor develops the various software products to be delivered under the contract.

a. Definition

Software engineering was defined earlier as:

> ...the technological and managerial discipline concerned with systematic production and maintenance of software products that are developed and modified on time and within cost estimates. [Ref. 3, p. 5-1]

b. Software Engineering Goals

The ultimate goal of the acquisition system is to field a system which meets the user's requirements. But because software is flexible, and can usually be modified more easily than hardware, those requirements evolve over time. This often results in a state of almost constant change for the system. The four goals of a software engineering discipline address the evolutionary nature of software systems: [Ref. 20, pp. 5-19 - 5-21]

- **Modifiability** - The ability to enhance, upgrade, and perform maintenance on software without increasing the complexity of the software's original design.
- **Reliability** - A critical determinant of software quality where the cost of failure is high as in weapon systems software. It is represented by probabilistic measures such as the mean-time-between-failure (MTBF) and mean-time-to-repair (MTTR). Reliability must be designed into the software, it cannot be achieved by testing after the fact.
- **Efficiency** - Efficiency is the highest and best use of critical resources. Of critical concern in real-time systems is throughput, which is the speed with which data can be processed and moved from one software module to another.
- **Understandability** - Understandable software reflects a natural view of the world. It is structured in such a way so that it is modifiable, efficient, and reliable.

c. Key Elements of Software Engineering

As shown in Figure 4, software engineering consists of the methods, tools, and procedures used to produce software. Software engineering occurs within the context of the contractor's software engineering environment (SEE). The SEE consists of the automated tools (either purchased commercially or developed internally) to augment the development process. It includes a test environment to perform qualification testing, a database that records all software development activities, and a software development library that facilitates the orderly development and subsequent support of the fielded software. [Ref. 20, p. 3-9]

SOFTWARE ENGINEERING ELEMENTS

Methods (How To's)	Tools	Procedures
Project Planning	Management	Sequence of activity
- Estimation	- Estimation	- Deliverables
- Scheduling	- Scheduling	- Controls
Requirements Analysis	Requirements	Inspection
Design	Design	Reviews
Coding	Development	Guidelines
Testing	Test	Standards
Maintenance	Documentation	
Measurement	Configuration	
Criteria for Quality	Management	

Figure 4. The Key Software Engineering Elements
From [Ref. 20, p. 5-18]

d. Software Development Models

Contractors may employ any number of different software development models. [Ref. 20, pp. 3-13 - 3-23] The evolutionary development model involves developing an operational product early, and then successively developing more refined versions. Incremental development involves developing the software product in groups of functional capabilities. This is characterized by a, build-a-little, test-a-little approach to deliver an initial subset of the final product. Prototyping allows the user to look at and evaluate various alternative software designs. This approach enables the user to determine if the software meets its requirements. The spiral model is a risk reduction methodology which incorporates the basics of the waterfall model (discussed below), and the incremental/evolutionary models.

While the waterfall model is the oldest of the models and is gradually being replaced by the others, it provides a simple model with which to describe the basic software development process. The waterfall model is depicted in Figure 5.

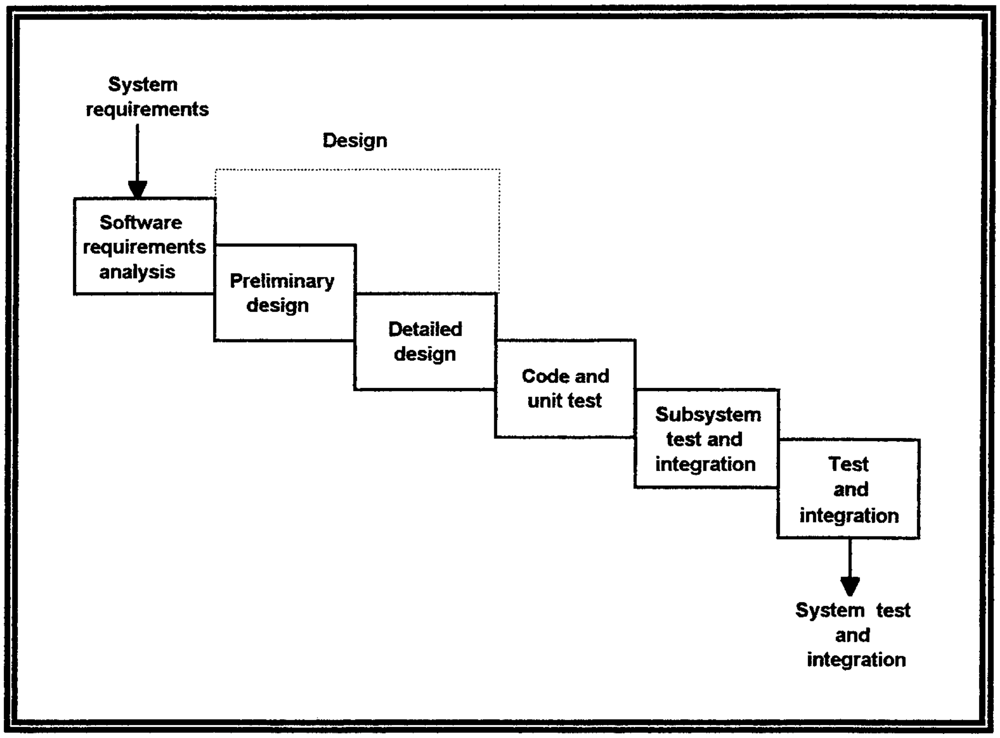

Figure 5. The Waterfall Development Model
From [Ref. 19, p. 21]

The following is a brief summary of the activities in each phase of the waterfall model: [Ref. 19, pp. 20-23]

- **Software Requirements Analysis** - In this phase the software system requirements are analyzed and allocated to the various Computer Software Configuration Items (CSCI). These are the major components, or subsystems defining the software architecture. At the end of this phase, software requirements are detailed enough to allow software design to begin.

- **Design** - The purpose of this phase is to map the requirements to the software architecture. In preliminary design, requirements are allocated to various Computer Software Components (CSC) which are distinct parts of a particular CSCI. In detailed design, the decomposition continues to the lowest level, the Computer Software Unit (CSU). This is a simple building block that can be traced back to the system requirements, coded or programmed by a single programmer in a short time, and is separately testable.

- **Code and Unit Test** - In this phase, individual CSUs are coded and tested. At this stage, testing is informal and usually conducted by the programmer.

- **Subsystem Test and Integration** - The purpose of this phase is to integrate the CSUs into increasingly higher levels of the software system until all major CSCs are tested.
- **Test and Integration** - At this point all CSCs have been integrated into a functional CSCI. The CSCI is then tested formally in accordance with the procedures agreed to in the contract. The purpose of the testing is to qualify the CSCI before the software system is integrated and tested in the actual weapon system.

e. Software Process Maturity

Contractors vary in the maturity and capability of their software development processes. Regardless of the specific development model employed, one of the key software engineering principles is that the process should be defined and in control. A tool which DoD Program Managers may use to measure the maturity of a contractor's software process is the Software Engineering Institute's (SEI) Software Process Maturity Model. The Model consists of 5 levels, and development at each level is better managed and organized than at lower levels. The levels are depicted in Figure 6.

Level	Characteristic	Key Problem Areas	Result
5 Optimizing	Improvement fed back into process	Automation	**Productivity & Quality**
4 Managed	(quantitative) Measured Process	Changing technology Problem analysis Problem prevention	
3 Defined	(qualitative) Process defined and institutionalized	Process measurement Process analysis Quantitative quality plans	
2 Repeatable	(intuitive) Process dependent on individuals	Training Technical practices Process focus	
1 Initial	(ad hoc/chaotic)	Project management Project planning Configuration mgt. Quality Assurance	**Risk**

Figure 6. The SEI Software Process Maturity Model From [Ref. 21, p. 5]

The SEI derived this empirical model from the collective experiences of many software professionals. One of the primary purposes of the model is to provide a framework for improvement of a contractor's software process. In this regard, the SEI found that the maturity model provides the following benefits: [Ref. 22, p. 93]

- It reasonably represents the historical phases of evolutionary improvement of actual software organizations.
- The model provides a measure of improvement that is reasonable to achieve from the prior level.
- It suggests interim improvement goals and progress measures.
- The model makes obvious a set of immediate improvement priorities once an organizations maturity level is known.

The maturity model is used in conjunction with a software process assessment method such as the SEI's Software Capability Evaluation (SCE). The purpose of the assessment is to determine the state of a contractor's software process. The SCE assesses the contractor's capabilities in four areas: (1) organization and resource management; (2) software engineering process and management; (3) tools and techniques, and (4) software development expertise. The SCE, together with the maturity model, serve two purposes for the PM. First, they provide a framework for the improvement of a contractor's process. Second, they may be used in the source selection of software developers. In this respect, the Air Force considers contractors below Level 3 on the maturity model to be high risk. [Ref. 20, p. 7-71]

This section has described the software acquisition environment in terms of the players involved and the three processes in which they operate to manage and develop weapon systems software. The next section addresses a particular issue in software acquisition management, the support or maintenance of fielded weapon systems software. What DoD call Post Deployment Software Support (PDSS).

D. SUPPORT OF DOD WEAPON SYSTEMS SOFTWARE

After a software-intensive weapon system has been developed, tested, and deployed, it enters the operations and maintenance phase of the system life-cycle. During this phase, it can be expected that the system will undergo numerous changes as the users identify deficiencies that need to be corrected, and as they request changes which enhance or add capabilities to the system. As shown in Figure 7, this phase of the software life-cycle accounts for a significant portion of the total life-cycle cost of the system.

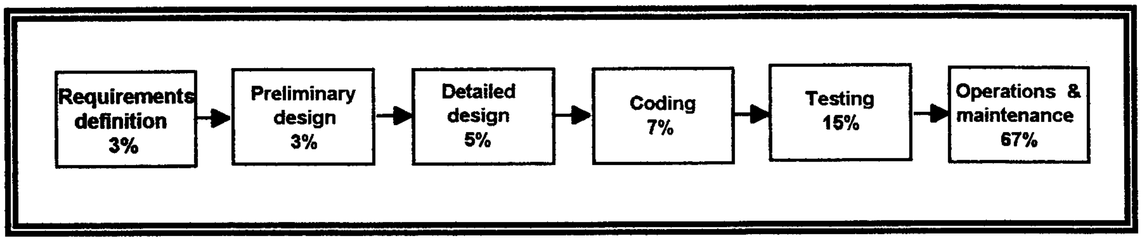

Figure 7. Software Life-Cycle Cost
After [Ref. 23, p. 4]

While this distribution of life-cycle cost makes software appear to be expensive, one of its benefits is that it can usually be modified faster and more cheaply than hardware. An example is the Army's PATRIOT Missile, which was originally developed to be an anti-aircraft weapon, but with modified software was able to also engage tactical ballistic missiles. Thus, the flexibility that software brings to modern weapon systems does provide a cost effective alternative to developing new systems to meet new or changing threats. [Ref. 24, p. 37] The process through which fielded weapon system software is maintained, supported, and evolves is called Post Deployment Software Support (PDSS).

1. Definition

Post Deployment Software Support is defined as:

> ...the sum of all activities required to insure that, during the production/deployment phase of a mission critical computer system's life, the implemented and fielded software/system continues to support its original operational mission and subsequent mission modifications and production improvement efforts. [Ref. 3, p. 7-5]

Throughout the software literature, the terms software maintenance, software support, and software evolution are used interchangeably. Regardless of the term used, in general, there are two categories of software maintenance activities: [Ref. 25, p. 9]

- **Corrective Maintenance** - This category focuses on fixing defects. Defects occur when the system does not perform as intended or as specified in the system requirements. This category historically accounts for only about 20 percent of software maintenance effort. [Ref. 26, p. 45]
- **Software Enhancements** - These are changes which are not corrective. There are two types of enhancements which together account for 80 percent of software maintenance effort.
  - Adaptive Maintenance - These are enhancements to accommodate changes in the operational or hardware environment. These account for 25 percent of effort. [Ref. 26, p. 45]
  - Perfective Maintenance - This category of changes improve software performance, maintainability, or other attributes. These account for 55 percent of maintenance effort. [Ref. 26, p. 45]

2. Software Maintenance Problems

Software maintenance brings with it a set of unique challenges. At the heart of these challenges is the issue of the "maintainability" of the software product that is delivered from the software development process. Essentially equivalent in definition to "modifiability" which was defined earlier, maintainability is defined simply as the ease with which the software can be changed to satisfy user requirements, and the effort required to add capabilities, correct deficiencies, or adapt to changing requirements. [Ref. 27, p. 13] Building in maintainability must be a consideration throughout software development and into the maintenance phase of the life-cycle. In a 1987 paper concerning software maintainability, Wilma M. Osborne of the National Bureau of Standards presented a list of the primary software maintenance problems which challenge the goal of maintainability. [Ref. 27] These problems and their corresponding characteristics are presented in Table 2.

PROBLEM	CHARACTERISTIC
Software Quality	- Design Quality - Computer Code Quality - Programming Languages Used - Increasing Inventory
Environment	- Growth - Evolving/Changing - Integrating New Hardware and Tools
Management	- Change/Configuration Management - Maintenance Techniques - Tools to: Restructure/Detect Errors/ Measure Code Complexity - Enforcing Software Standards
Users	- Demanding More Capabilities
Personnel	- Lack of Experience - Morale Problems - View of Maintenance: Unchallenging/Unrewarding

Table 2. Software Maintenance Problems After [Ref. 27, p. 14]

As is the case with software development in general, Osborne finds that while there are both technical and managerial problems with software maintenance, ***management*** of the process is the key issue.

3. PDSS Policy

In an a attempt to better manage maintenance of weapon systems software, the Army has developed a draft PDSS policy issued by the Director of Information Systems for Command, Control, Communication and Computers (DISC4). [Ref. 28] The highlights of the policy are listed below:

- **Planning and Preparation** - Planning for PDSS must begin at the beginning of the system life-cycle, and be documented in the Computer Resources Life-Cycle Management Plan (CRLCMP). To ensure that the system will be supportable, the assigned organic Software Support Activity (SSA) must complete a Software Supportability Assessment at each milestone.

- **Continued Support** - PDSS of the system will continue until the system is removed from the inventory or earlier if so determined by the Deputy Chief of Staff, Operations (DCSOPS). Cost-benefit analysis will be used to determine when software should be upgraded or re-engineered, and to determine the level of support.

- **Centralized Priorities and Funding** - PDSS is a Department of the Army (DA) directed program for prioritization and funding. PDSS funding will be controlled centrally by DA based on prioritized input from the Major Commands. PDSS actions will be prioritized and categorized as either:

  - Minimum Essential (Band 1) - This band is divided into two bands: Minimum Sustainment (Band 1.1) - Those activities required to maintain minimum essential capabilities; and "Must Do" Requirements (Band 1.2) - Changes required to correct problems which impact mission essential capabilities.

  - "Should Do" Requirements (Band 2) - Those activities which will enhance system performance or add capabilities.

- **Government Managed** - Government personnel are responsible for monitoring and guiding all of the various tasks throughout the system life-cycle. In deciding whether PDSS will be performed by a contractor, the Government, or a combination of both, the Program Manager will develop a "business case analysis" which will be coordinated with the Program Executive Officer (PEO) and supporting command. If contractor support is selected, the program's assigned SSA must perform a Supportability Assessment. If Government support is selected, the SSA and Program Manager must jointly develop a transition plan.

4. PDSS Guidance

The principal DoD "how to" guidance concerning PDSS is given in Military Handbook 347, Mission-Critical Computer Resources (MCCR) Software Support (MIL-HDBK 347). [Ref. 25] The MIL-HDBK provides guidance concerning the entire PDSS process. This includes both the planning and preparation for PDSS, and how to conduct software support.

a. Software Support Principles

The MIL-HDBK provides the following software support principles: [Ref. 25, pp. 12-13]

- Software support activities occur throughout the system acquisition process.
- The evolution of software is a continuum of successive software development cycles beginning with initial development and continuing with one or more follow-on development cycles.
- Software should continue to evolve as long as essential requirements or cost-effective benefits can be derived through software change.
- The SSA is the Government's agency responsible for providing PDSS requirements to the Program Manager and for cost effective PDSS operations following transition.
- Critical MCCR activities are performed by the Government to ensure accountability, control, continuity, and compliance. Critical activities include: control and identification of software baselines; defining and evaluating software quality requirements; system level testing; and planning and managing PDSS.
- Direct communication between the user and the SSA is important to help identify and isolate software problems, and foster support and cooperation.

b. Pre-Deployment Software Support

Recall that one of the primary activities involved in the acquisition management process is planning. As such, a program's system of software support must be based on sound planning. Pre-Deployment Software support are those activities which occur prior to a system's initial fielding. They include: (1) Planning for PDSS; (2) Identifying PDSS acquisition requirements; (3) Ensuring software supportability; (4) Assuring software quality, and; (5) Development and implementation of a transition plan. Each of these activities is addressed briefly below: [Ref. 25, pp. 18 - 24]

- **Plan for PDSS** - The Computer Resources Life Cycle Management Plan (CRLCMP) is the primary document for computer resource planning and management throughout the system life cycle. One of the CRLCMP's primary purposes is to document the PDSS concept. The PDSS concept is the basis for all PDSS planning and is derived from the overall system Integrated Logistic Support (ILS) concept. It describes how and to what extent the software will be supported during deployment. If PDSS planning begins early, the set of feasible PDSS options will be larger than if planning is done late. In the absence of a clear PDSS concept, decisions are made which limit future choices concerning how the system will be supported. A second purpose of the CRLCMP is to document the resources required by the SSA to implement the PDSS concept. These resources include, facilities, the software engineering environment, and personnel.

- **Identify PDSS Acquisition Requirements** - The PDSS acquisition requirements are those contractual requirements imposed on the software developer which will facilitate PDSS. These requirements may include use of specific software engineering tools or environments, technical data requirements, quality requirements, involvement of the SSA in software development, and transition issues.

- **Ensure Software Supportability** - Specifying software support requirements in a contract does not ensure software supportability. The SSA has a vested interest in ensuring supportability through active involvement in the software acquisition process. This should include evaluation of the software product's supportability. While somewhat subjective, supportability takes into account the following: the characteristics of the software product, its completeness, correctness, and quality; the software development environment; and the resources required to support the software. In ensuring software supportability, the SSA may: participate in technical reviews and audits; participate in formal qualification testing; or perform Independent Verification and Validation (IV&V) (the process by which the requirements and correctness of the software are checked by someone other than the developer).

- **Assure Software Quality** - Like supportability, quality is not something that can be assured by specification alone. It requires the SSA to be actively involved in activities such as authenticating specifications, verifying requirements, and evaluating proposed quality plans and records.

- **Develop and Implement the Transition Plan** - Software transition includes all activities required to transition the software development capabilities from the software developer to the SSA. It involves the transfer of computer hardware, software, data, and experience. The primary focus of the transition is the smooth and effective transfer of the Software Engineering Environment (SEE) and the Software Test Environment (STE), to allow the SSA to begin supporting the software. Software transition is a major undertaking which requires detailed planning. Proper planning is essential, without it, transition is inefficient. Dr. Thomas Vollman provides examples of the transition problems caused by poor planning. [Ref. 29, pp. 191 - 192] These may include a lack of adequately trained staffing at the support activity, the lack of defined procedures to keep the software and the process under control and a significant loss of corporate knowledge of the system. In a worst case scenario, the SSA will not be able to provide PDSS. Important transition activities that must be addressed include:

- Training.
- Staffing.
- Turnover, installation, checkout, and integration of any hardware or software acquired from sources other than the software developer.
- Implementation of required PDSS capabilities. This may include problem replication and fault isolation, corrective action, software development, documentation, and integration and test.
- Approval and implementation of software configuration and quality management plans.
- Verification that all transition milestones have been completed and that all required resources are available.
- Integration of all PDSS activities into a cohesive PDSS process.
- Reporting transfer of software product.
- Determination that security requirements have been met.

In summary, half of the software support battle is in the proper, early planning of a PDSS concept. If a transition to the Government is contemplated, thorough planning and preparation is necessary.

c. The PDSS Process

As is the case with software development, software maintenance requires a structured approach in order to be effective. Similar structured approaches are outlined in both the IEEE Standard for Software Maintenance, IEEE Std 1219-1993 [Ref. 30], and in Military Handbook 347, Mission Critical Computer Resources Software Support. The general PDSS process is depicted in Figure 8. The following paragraphs provide a brief discussion of the five major phases: [Ref. 25, pp. 25-27]

- **Initial Analysis Phase** - The input to the PDSS process is a change or problem report. These may be generated by the user or by programmers working on the system. The SSA receives the problem report and provides information to the Configuration Control Board (CCB). This includes a detailed description of the change, the classification of the change, analyses of the management, technical, and support impacts of the change, the estimated effort required to implement the change, and the risks associated with implementing the change. The CCB has the authority to make the decision on a change.

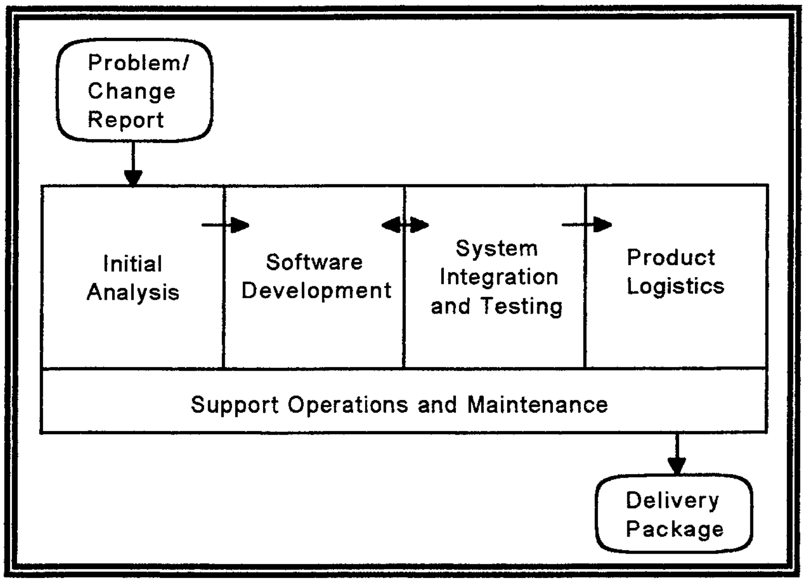

Figure 8. The General PDSS Process
From [Ref. 25, p. 25]

- **Software Development Phase** - After the CCB makes the decision to implement a change, it can be developed and coded by whatever process is in place at the support activity. This phase includes the analysis of the requirements; preliminary and detailed design of the change; and the coding and testing of the particular software configuration item(s) (CSCI) impacted by the change.

- **System Integration and Test Phase** - The purpose of the integration and test phase is to incrementally build the change and to logically test the developing system. During this phase, CSCIs are integrated and tested to facilitate the isolation and correction of errors. This phase also includes testing of the changed software on the target hardware - the weapon system's computer which runs the software. If the test version of the software is approved, the configuration identification is updated and the process begins again.

- **Product Logistics Phase** - The product logistics phase is the final PDSS phase and involves the logistic support required for the new operational software version which has been created. This may include user training and site installation and checkout. The final product of the PDSS process is a delivery package in the hands of the trained user and a configuration that matches the approved software version.

- **Support Operations and Maintenance Phase** - Support operations occur throughout the PDSS process and include activities such as: computer center operations, tactical systems operations, and security.

5. Software Support Summary

With so much of the life-cycle cost of weapon systems software concentrated in the operations and maintenance phase of the system life-cycle, software maintenance or support is clearly an area which deserves thorough planning, and management attention. Author and software maintenance expert Thomas M. Pigoski gives the following perspective on the importance of software support: [Ref. 31, p. 11]

- Functions are migrating to software - from hardware to software and from manual to software.
- One reason that functions migrate to software is because it is soft - changeable, flexible, modifiable.
- The sustained, incremental, controlled modification of software (the enhancement and adaptation activities of software support) is therefore integral to the effectiveness of software.
- Software support requires high degrees of professionalism, technique, and technology.

In regard to these principles he states:

> Yet these propositions themselves were and are in a state of change. The migration of functions to software is an ongoing process whose scope and limit we cannot yet see. The softness of software is only relative, and poses significant dangers and problems of its own. The acceptance of software support as a key to the effectiveness of software is a social change that still provokes controversy. And the technical and managerial environment for software support is something that researchers, educators, and vendors are still developing. [Ref. 31, p. 11]

E. CHAPTER SUMMARY

Software acquisition has historically been and continues to be a challenging activity for the DoD Program Manager. The software acquisition process is a complex interaction of three interrelated processes: the acquisition management process, the project management process, and the software engineering process. The product of the acquisition process is weapon systems software. Because it is modifiable, it continues to evolve throughout the life of the weapon system to meet new user requirements and new threats. Software maintenance accounts for about 70 percent of the life-cycle cost of software, and more and more weapon systems software is being acquired. Post Deployment Software Support is a vitally important part of the software acquisition management process.

III. CASE DESCRIPTION: POST DEPLOYMENT SOFTWARE SUPPORT OF THE SPECIAL OPERATIONS AIRCRAFT

The President has ordered the cancellation of an operation in Iran which was under way to prepare for a rescue of our hostages. The mission was terminated because of equipment failure. During the subsequent withdrawal of American personnel, there was a collision between our aircraft on the ground at a remote desert location in Iran. There were no military hostilities, but the President deeply regrets that eight American crew members of the two aircraft were killed and others were injured in the accident....The nation is deeply grateful to the brave men who were preparing to rescue the hostages....

White House Statement, 1 a.m., April 25, 1980

A. INTRODUCTION

This official account of the failed Iranian hostage rescue mission, along with the images of the charred remains it left behind at "Desert One," dramatically highlighted the need for military aircraft which are specially designed and built to withstand the rigorous demands of the Special Operations Forces (SOF) aviation mission environment. This environment is typified by the conditions encountered during the failed mission. It requires the capability for SOF aviation to conduct clandestine airlift missions under conditions of limited visibility, any type of weather, and over extremely long-ranges. The Army's Special Operations Aircraft (SOA) is part of DoD's solution to the SOF aviation challenge.

This chapter provides the background of the U.S. Army Special Operations Aircraft (SOA) Program and a case description of the elements of the SOA Program's current Post Deployment Software Support (PDSS) acquisition process. The chapter is divided into two major sections. The first section provides the SOA Program background, to include a summary of the mission need, the program acquisition strategy, and a summary of the Integrated Avionics Subsystem (IAS) software development. The second major section provides the description of the SOA Post Deployment Software Support case. It provides a description of the software system being supported by PDSS, the original PDSS plan and how the plan was modified by the SOA PDSS Process Action Team (PAT). It also includes a discussion of the key players involved in the PDSS effort, a description of the PDSS acquisition process, and a summary of the activities accomplished to date under the base year, and first option years of the PDSS contract.

B. SOA PROGRAM BACKGROUND

1. The Mission Need

The Special Operations Forces Helicopter Modification (SOF MOD) of the UH-60L and the CH-47D aircraft to the MH-60K and MH-47E configurations was initiated by the Department of the Army in April 1986. It was in response to the DoD Special Operations Forces Airlift Report and the Special Operations Forces Required Operational Capability (ROC), both of which are classified SECRET. The SOF MOD Project called for the design, integration, modification, and qualification of a Mission Equipment Package (MEP). This included an Integrated Avionics Subsystem (IAS) to be integrated into both the UH-60L BLACK HAWK, and the CH-47D CHINOOK airframes. The purpose of the MEP is to enhance an aircraft's capability to operate reliably in the demanding SOF mission environment and to reduce pilot workload and stress. The SOF MOD project used a Non-developmental Item (NDI) acquisition approach and was originally designated a nonmajor system. The Project Management Office (PMO) was initially established simply as a matrix organization within the engineering directorate at the U.S. Army's Aviation Systems Command (AVSCOM).

As indicated earlier, the project was the direct result of the deficiencies in the current fleet of standard configuration aircraft to meet the needs of the SOF aviation mission. SOF aviation doctrine calls for aircraft which are capable of conducting clandestine, deep penetration, low-altitude airlift missions, over all types of terrain, under adverse weather conditions, with little or no visibility. Additionally, these aircraft must have increased protection from threat anti-aircraft weapon systems, as well as increased self-deployment capability. To meet these requirements the SOA MEP includes: [Ref. 32]

- An IAS to enhance communications and navigation;
- Improved Aircraft Survivability Equipment (ASE);
- Increased armament which includes suppressive weapons;
- The addition of external and internal fuel tanks, and air-to-air refueling probes;
- Upgraded engines; and
- An upgraded transmission on the MH-60K aircraft.

The configurations of both aircraft are shown in Figures 9 and 10.

2. SOA Program Acquisition Strategy

The Product Manager, Special Operations Aircraft (PM SOA) was officially established and chartered in September 1987. As an NDI acquisition based on proven airframes and avionics equipment, the acquisition strategy did not include a Demonstration and Validation Phase. As shown in Figure 11, the SOA Master Program Schedule, a Milestone II In-Process Review (IPR) was conducted in October 1987 which authorized the program to proceed with development of one prototype aircraft for each airframe. The program acquisition strategy included the award of letter contracts for prototype research and development in January 1988. The Airframe Contractors (ACs) were Boeing for the MH-47E and Sikorsky for the MH-60K Because cost control was a major priority, [Ref. 32] these contracts were later definitized as firm-fixed-price contracts. The IAS contractor, IBM Federal Systems Division (now LORAL Federal Systems) was selected by a Government competitive Source Selection Evaluation Board in August 1987. However, the Government did not contract directly with IBM. Rather, it became a directed subcontractor to the ACs. The modifications developed by the ACs, who had total system performance responsibility for integration of the MEP into the respective airframes, were planned to be incorporated into the standard aircraft by an Engineering Change Proposal (ECP).

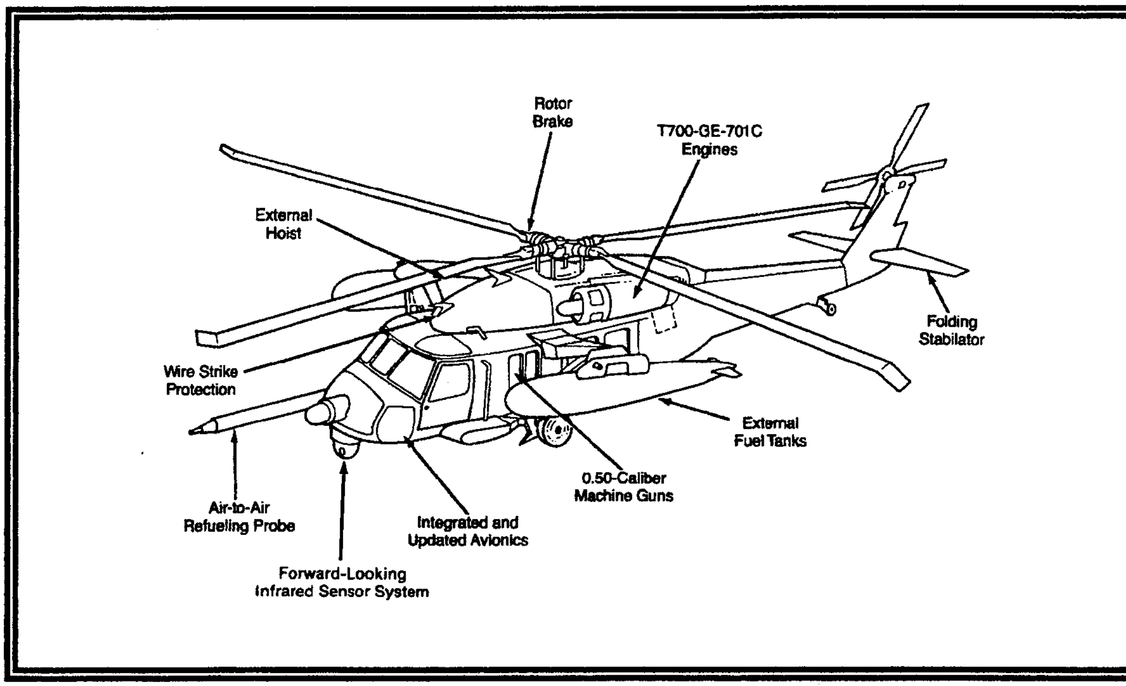

Figure 9. The MH-60K Configuration
From [Ref. 33, p. 14]

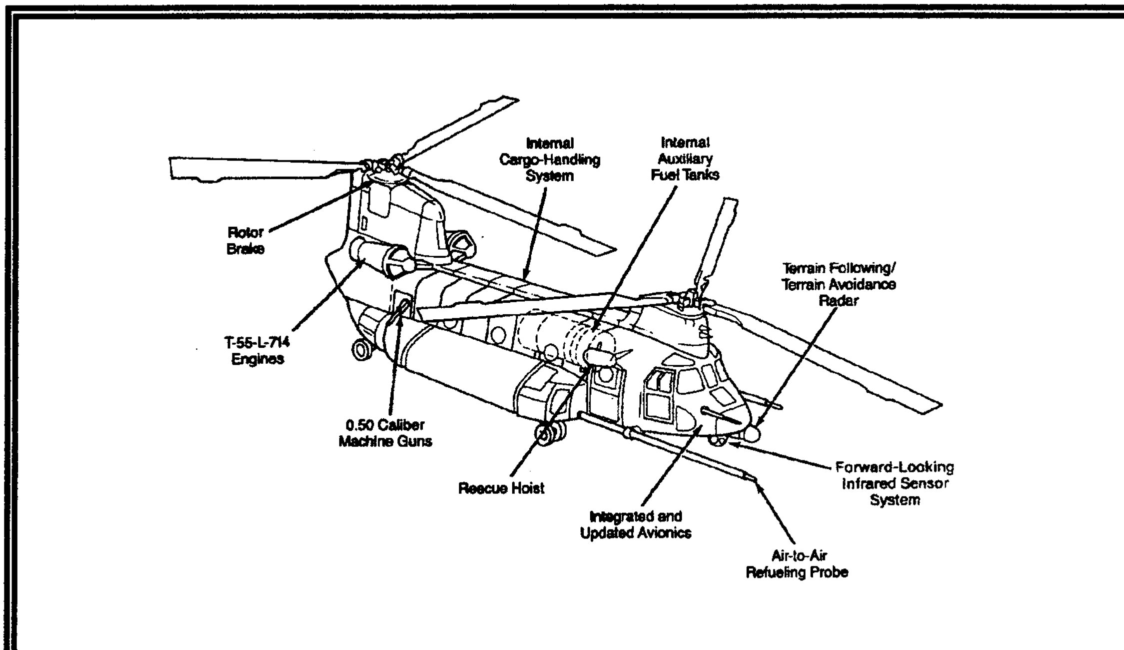

Figure 10. The MH-47E Configuration
From [Ref. 33, p. 13]

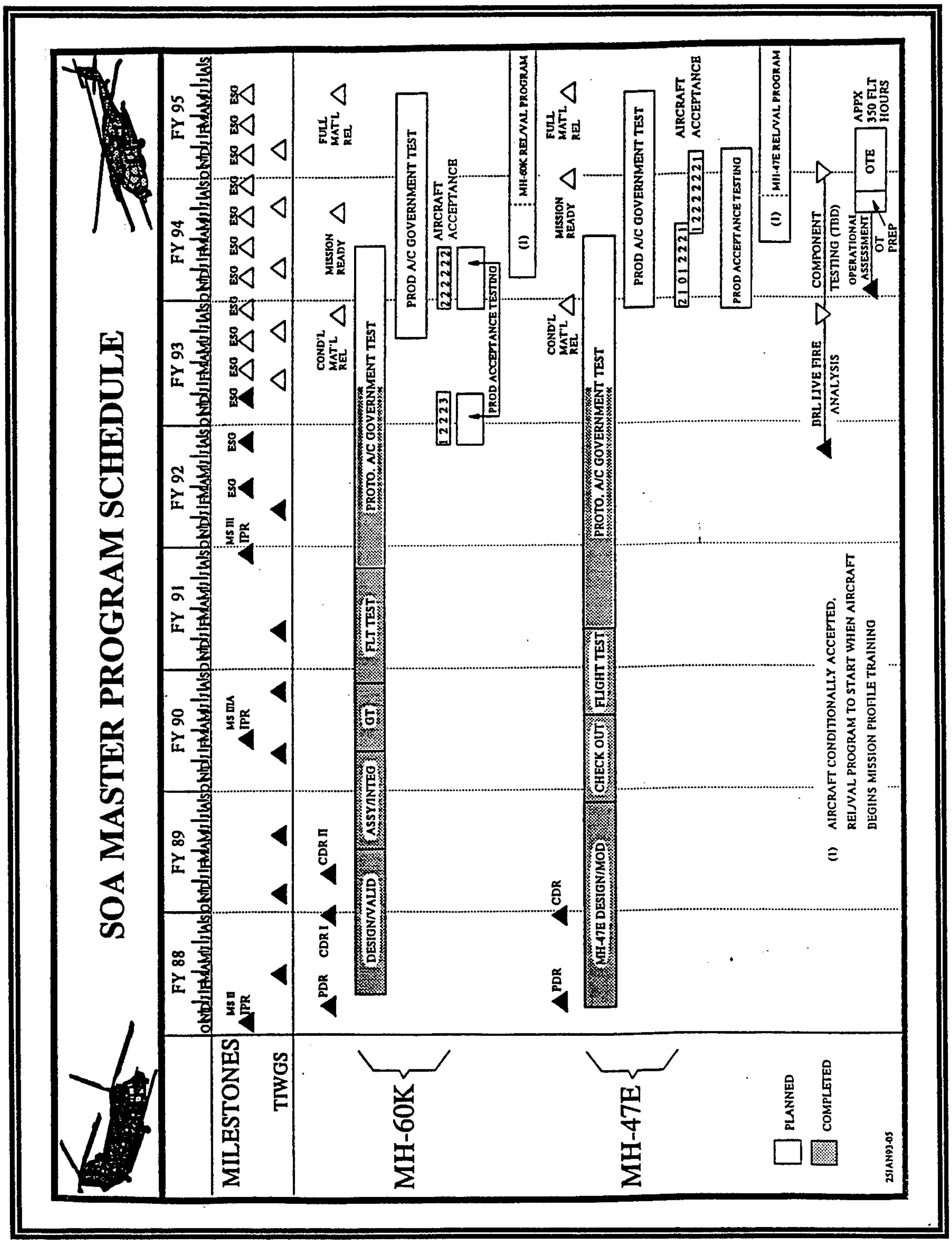

Figure 11. SOA Master Program Schedule
From [Ref. 6]

DoD Instruction 5000.2 defines Non-developmental items as those which do not require development. This includes those items which require only *minor* modification to meet the agency's need. [Ref. 13, p. 6-L-1] In this respect, SOA was considered an NDI acquisition because the airframes and each of the avionics components to be integrated into them had been previously qualified. Despite the SOA program's NDI status, it was not without its risks. The primary source of program risk was that it required considerably more "development" than the NDI status had anticipated. However, the Army considered the program risks to be manageable. A June 1990 letter from the Secretary of the Army to Congress outlined the major areas of risk involved in the SOA program. These included: [Ref. 34]

- The processing capacity of the IAS Mission Processor, AP-102. This was considered a medium cost risk. The worst case scenario judged that the processor would operate at 93 percent capacity versus the 80 percent goal.
- Optimization of the flight algorithm for the Multimode Radar, AN/ALQ 174 which provides the aircraft's terrain following/terrain avoidance capability. This was judged to be a low risk item, but the Government would be at risk to conduct additional development if optimization could not be achieved within the three months of contractor flight testing allocated for the effort.
- Integration of Government Furnished Equipment. This was considered a low to medium cost risk.
- Retrofit of four production aircraft as a result of concurrent prototype development and production. This was considered a low to medium cost risk.

While mentioned as a low to medium cost risk, in hindsight, the concurrency of development and production was in fact one of the major sources of risk in the SOA acquisition strategy. This was a characteristic for which the program was criticized by the GAO in a 1990 audit report. [Ref. 33] The basis of the criticism was that despite the fact that the modification program required the integration of about 40 items (these included communications, ASE, and navigation equipment) the development of the prototype aircraft was not completed before the Army approved a Low Rate Initial Production (LRIP) decision at Milestone IIIA, in February 1990. The LRIP decision authorized the production of 11 aircraft of each type. The DoD response to this criticism [Ref. 35] centered on these facts:

- That because SOA was not a major acquisition program, regulations did not require the program to undergo formal Initial Operational Test and Evaluation (IOT&E) prior to a production decision.
- Operational testing in the program consisted of Early User Test and Evaluation (EUT&E) conducted by user pilots and mechanics in conjunction with contractor prototype tesing.
- The SOA testing program was in consonance with NDI acquisition regulations.
- Each of the avionics components to be integrated into the IAS had been previously qualified, or were currently in use on other DoD systems.

However, at least some of the GAO's concurrency cautions proved true. Later cost growth elevated the SOA program from non major program status to a Category II Major Program in January 1992. [Ref. 6] Growth in the IAP software required an upgrade in the Mission Processor to the AP-102A configuration. And in December 1993, as one of the program's contracting officers, the researcher personally awarded a contract directly to IBM to complete development and qualification of the terrain following (TF) capability of the Multimode radar (MMR). [Ref. 6, pp. 2-12 - 2-13] Despite these and other challenges encountered during the highly concurrent program, the production quantity of 22, MH-60K and 25, MH-47E aircraft have been delivered. The user, the 160th Special Operations Regiment (Airborne) now has the capabilities originally identified in the aftermath of "Desert One." By Fiscal Year 1994, these capabilities cost a total of $1.38 billion. [Ref. 6] The development and production schedule for the program spanned only eight years. Support of both aircraft is provided by a Life-Cycle Contractor Support (LCCS) Joint Venture of both ACs and named Boeing-Sikorsky Air Services.

3. The IAS Software Development

For one who has been familiarized with the difficulties involved with weapon systems software development, it is not unusual that development and integration of the IAS and its software proved to be one of the greatest challenges involved in the program.

Over a four year software development, 12 iterations of the Integrated Avionics Program (IAP), (the "brain" of the IAS) were built before the system performed in accordance with the Prime Item Development Specification (PIDS). In reviewing the history of the program, several factors contributed to this problem. First was the cancellation of the V-22 Program in November 1989. The SOA program was to have benefited from the V-22 as the lead program in the IAS development. A second factor was the disagreement between the ACs, IBM, and the Government over exactly what the PIDS required. This was especially questioned because both the development and production contracts were firm-fixed-price. A third factor concerned a lack of communication between the Government and the SOA contractors.

In an April 1993 interview, the SOA Product Manager (Lieutenant Colonel Michael W. Rogers) described the Program's software problem as follows: [Ref. 36, p. 40]

> We were told not to worry about the software, that the next iteration of software would solve all the problems we had been experiencing in the past. What we had been told was incorrect. The main problem we faced with the two helicopter programs was that there was little communication between the Government and the contractors. We had no way to resolve the software trouble reports (STRs). There was no process of keeping track of the software problems we were having.

To address the software issue, Lieutenant Colonel Rogers formed what he called, Team SOA. Team SOA was a Government-contractor team which consisted of the following elements:

- An executive steering group whose membership included Army leadership at the general officer level and the contractor CEOs;
- A management working group consisting of the Government and contractor PMs; and
- Several process action teams which addressed specific software problems.

Also instrumental to the resolution of the STR problem was user involvement in the software development process. This ensured that STRs were resolved to the user pilot's satisfaction. Figure 12 depicts the STR resolution trend experienced throughout software development. This trend evidences the complexity of the IAS software development. It shows that by the time production software version 10.3.2 was delivered, nearly 5,000 STRs had been identified. It also evidences the effectiveness of the corrective action, since the number of remaining STRs was later reduced to about 62. [Ref. 37]

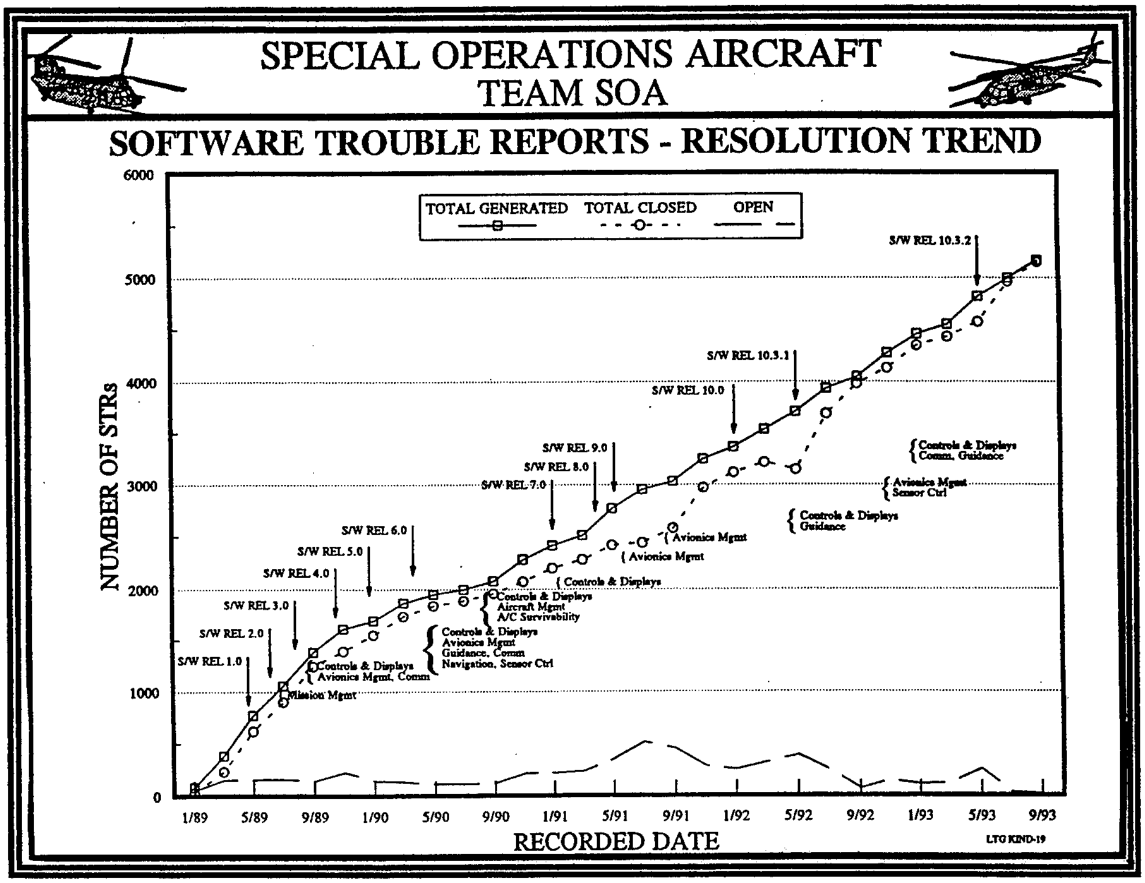

Figure 12. Software Trouble Reports-Resolution Trend From [Ref. 38]

This section has provided a brief description of the background of the SOA program. It highlighted the risks encountered during development and the complexity of the IAS software development. The next section describes how the IAS software is supported.

C. SOA PDSS CASE DESCRIPTION

1. The Supported Software System

This section describes the major software-intensive components of the SOA system. These include the IAS, non-IAS components such as the MMR, and training devices.

a. The SOA Integrated Avionics Subsystem

The heart of the capability of the MH-60K and MH-47E Special Operations Aircraft, and the focus of the Post Deployment Software Support (PDSS) of the aircraft, is the Integrated Avionics Subsystem (IAS). The IAS hardware and software are common to both the MH-60K and the MH-47E aircraft. Their primary purpose is to enhance mission management and reduce crew workload. The IAS consists of 15 Line Replaceable Units (LRU) or "Black Boxes." These contain approximately 380,000 SLOC in nine CSCIs, which run on ten separate computer processors. The IAS and the other non-IAS mission aids are integrated on the aircraft using two, dual-redundant MIL-STD-1553B data buses.

The IAS performs four major functions. These are: [Ref. 39]

- Navigation - maintaining the aircraft's current state;
- Guidance - determining the reference path to the next point in the mission plan;
- Flight Director - providing steering cues to the reference path; and
- Mission Management - planning the entire desired mission.

A representation of the IAS is given in Figure 13.

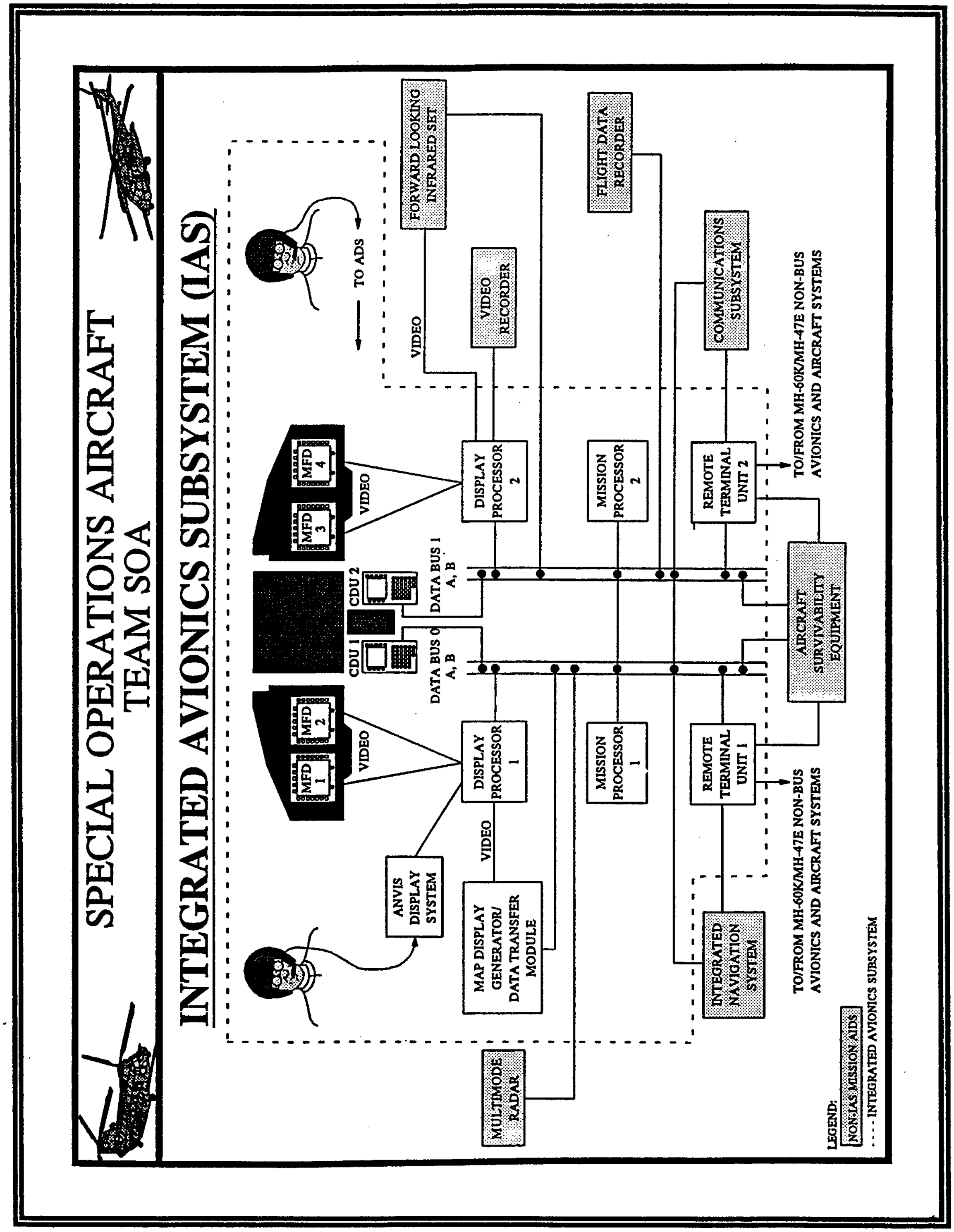

Figure 13. The SOA Integrated Avionics Subsystem
From [Ref. 6, p. 2-6]

The IAS is divided into two major areas, the Mission Management System and Mission Aids. It consists of the following LRUs in the quantity indicated: [Refs. 6, 7, 39, 40]

- Mission Management System (MMS)
  - Mission Processor, AP-102A Computer (MP) (2)
  - Display Processor (DP) (2)
  - Color Multifunction Display (CMFD) (2)
  - Monochromatic Multifunction Display (MMFD) (2)
  - Control Display Units (CDU) (2)
  - Remote Terminal Units (RTU) (2)
- Mission Aids
  - Map Display Generator (MDG)/Data Transfer System (DTS) (1)
  - Aviator's Night Vision Imaging Systems (ANVIS) Display System (1)
  - Grip Assembly, Tracking Handle (1-MH-60K, 2-MH-47E)

b. Non-IAS Mission Aids

In addition to the IAS, the following non-IAS mission aids are also integrated over the 1553B data bus: [Ref. 41]

- Multimode Radar - Provides the capability for low-level, terrain following (TF) flight in limited visibility.
- Forward Looking Infrared (FLIR) Sensor - Provides magnified night vision capability.
- Integrated Navigation System - Provides a broad spectrum of sophisticated sensors which allow highly accurate navigation to and from the mission area. The system includes a Global Positioning System (GPS), Inertial Navigation Unit (INU), Doppler Navigation Unit, and a Personnel Location System.
- Aircraft Survivability Equipment (ASE) - Provides early warning and countermeasures against threat anti-aircraft weapons.
- Communications Subsystem - Provides secure communications, and command and control capability with a broad spectrum of radio systems including SATCOM, HF, and SINCGARS.

- Flight Data Recorder and Video Recorder - Allows crews to record and review data transmitted over the 1553B data bus, and the FLIR sensor.

c. IAS Software

The IAS software is resident in nine Computer Software Configuration Items (CSCIs). The reader should note that the two CSCIs resident in the mission processor (MP) are LORAL products, while the remaining CSCIs were developed by subcontractors. These are shown in Figure 14 and an explanation is provided in the paragraphs below: [Ref. 7, pp. 3-7]

- Integrated Avionics Program (IAP) CSCI - The IAP provides the overall system control and is, in essence, the "brain" of the aircraft. It interfaces and integrates information from the more than 35 other subsystems on the aircraft. These include: engines, transmissions, MMR, FLIR, communications, navigation, and ASE. It is the first of two CSCIs which run in the mission processor (MP).
- Start-Up Read Only Memory Program (SROMP) CSCI - The SROMP provides the ability for the MP to be loaded over the 1553B data bus. Since the IAP is identical on both airframes, the SROMP checks the aircraft wiring during start up and identifies to the IAP and MP which type of aircraft it is on. The SROMP executes on the MP but is stored as firmware, a combination of hardware and software.
- Display Processor (DP) CSCI - Each DP drives two Multifunction displays. The DP controls video inputs, construction and display of symbology, and timing/interface with the 1553B data bus. It is partitioned into two functions:
    - Display Management Program (DMP).
    - Symbol Generator Program (SGP).
- Display Processor Memory Loader (DP ML) CSCI - Provides memory load/verification capability over the 1553B data bus and performs the Power-On Built-In-Test (PBIT) for the DP.
- Remote Terminal Unit (RTU) Operational Flight Program (OFP) CSCI - The Remote Terminal Unit provides the interface conversion between the MP and the non-1553B data bus compatible avionics components.
- Remote Terminal Unit Memory Loader (RTU ML) CSCI - This CSCI provides memory loader/verification capability over the 1553B data bus and performs the PBIT function for the RTU.

SPECIAL OPERATIONS AIRCRAFT TEAM SOA

SOA SOFTWARE OVERVIEW

CSCI	MAJOR FUNCTION	SOURCE LINES OF CODE		DELTA % (88-93)	LANGUAGE	DEVELOPER	CPU USED
		1988	1993				
IAP	IAP	68772	196641	188	JOVIAL ASSEMBLY	IBM	MIL-STD 1750A
SROMP	MP STARTUP		1754		ASSEMBLY	IBM	1750A
DP	DMP	61560	84775	105	JOVIAL ASSEMBLY	ALLIED	1750A
	SGP		39419		ASSEMBLY	ALLIED	INTEL 8751
DP ML	ML		1843		JOVIAL ASSEMBLY	ALLIED	1750A
RTU	OFP	5260	13327	165	ADA ASSEMBLY	ALLIED	1750A
RTU ML	ML		632		ASSEMBLY	ALLIED	1750A
MDG	MMP	7200	14830	468	ADA ASSEMBLY	ALLIED	1750A
	DCE		25211		ASSEMBLY	ALLIED	TMS 320C25
MDG ML	ML		892		ASSEMBLY	ALLIED	1750A
CDU	CDU	4000	4842	21	ASSEMBLY	SMITH	INTEL 8031
MFD	DISP CTR	1821	2850	57	ASSEMBLY	ALLIED	INTEL 8031A
		148613	387016	160%			

LTG KIND-20

Figure 14. SOA Software Overview
From [Ref. 38]

- Map Display Generator (MDG) CSCI - The MDG CSCI provides the advanced map display capabilities. It uses a modified Defense Mapping Agency (DMA) Digital Landmass database, aeronautical chart data, and digitized data to produce a map display which can be overlaid with mission planning data. It consists of two major functions:
  - Map Management Processor (MMP).
  - De-Compression Engine (DCE).
- Map Display Generator Memory Loader (MDG ML) CSCI - This CSCI provides a memory load/verification capability over the 1553B data bus and performs the PBIT function for the MDG.
- Control Display Unit (CDU) CSCI - The CDU provides the pilot/hardware interface. Using the CDU keyboard, the crew are able to modify the IAS data or update mission related information such as navigation waypoints or communications frequency changes.
- Multifunction Display (MFD) - While not considered a CSCI, the firmware resident in each MFD responds to the output generated by its corresponding DP. The 24 push buttons around the MFD bezel provide the pilot control interface with the IAS.

d. Training Devices and Support Equipment

The Special Operations Aircraft system also includes the following software intensive training devices and support equipment. They must also be included when considering PDSS: [Refs. 6, 39]

- Desk Top Trainer (DTT) - Trains pilots and maintainers in the use of the SOA Mission Management system, and operation and control of the Integrated Avionics Program (IAP) through the multifunction display. It runs a CSCI of 24 thousand SLOC in a desktop personal computer.
- Part Task Trainer (PTT) - Trains pilots and maintainers through interactive simulation of the MH-60K and MH-47E equipment using "Touch Screen" technology. It runs two CSCIs of 54 thousand SLOC in the earlier version of the mission processor, the AP-102.
- Combat Mission Simulator (CMS) - A full motion simulator for each aircraft located with the user at Fort Campbell. It was developed and is supported by the U.S. Army Simulation, Training, and Instrumentation Command (STRICOM).

- Ground Support System Software Load Device (GSS SLD) - An IBM ThinkPad laptop computer which connects to the 1553B data bus to gather maintenance data and to load software into IAS processors.

2. PDSS Plan

The reader will recall from Chapter II. that one of the keys to effective PDSS is prior planning. Such planning is properly conducted during what MIL-HDBK-347 calls Pre-Deployment Software Support. It includes activities such as: (1) planning for PDSS; (2) identifying PDSS acquisition requirements; (3) ensuring software supportability; (4) assuring software quality, and; (5) development and implementation of a transition plan. [Ref. 25, pp. 18-24] Each of these activities should be addressed in the weapon system's Computer Resources Life Cycle Management Plan. This document is developed by the PM, in coordination with the Government Software Support Activity (SSA) that is designated to perform PDSS for the system after fielding.

Throughout the development of the SOA IAS software, the assigned SSA was the Software Engineering Directorate, Avionics Branch, at the U.S. Army Communications-Electronics Command (CECOM). CECOM software engineers were most closely involved in the software development effort through the conduct of Independent Verification and Validation (IV&V) of the software design documentation, and code listings. In preparation for the transition of the software engineering environment (SEE) from LORAL to CECOM, one of the system's major components, the Army SOA Helicopter Avionics System Simulator (AHASS) was re-hosted from a mainframe computer to a smaller, workstation system. [Ref. 42] The SSA's plan for software support, and a plan to transition software development responsibilities from LORAL to the SSA were included in the SOA Computer Resources Management Plan (CRMP). The highlights of this plan were: [Ref. 7]

- The purchase of a complete Software Engineering Environment (SEE) to maintain all of the IAS CSCIs. The plan called for 18 months to procure and install the SEE.

- The conduct of PDSS training through participation in software development activities at LORAL's facility, followed by a period of several years during which LORAL's involvement in PDSS would be gradually phased out.
- Transition of PDSS responsibility from LORAL to the SSA 120 days prior to the fielding of the last production aircraft.
- Process flow charts which addressed: the flow of STRs; the configuration management process; and the Engineering Change Proposal Process (ECP).
- Annual software releases for the IAP during PDSS.

It should be noted, that while the typical Government SSA is Government managed, the majority of the software development work is conducted by support contractor personnel. This is true not only at CECOM, but is a DoD wide challenge and was witnessed by the author at two other DoD SSAs. [Refs. 43, 44] The CECOM estimates for man-years (MY) of PDSS labor, and PDSS funding in millions of dollars are shown in Tables 3 and 4 respectively.

PDSS RESOURCES	**FY92**	**FY93**	**FY94**	**FY95**	**FY96**	**FY97**
Support Personnel:	**MY**	**MY**	**MY**	**MY**	**MY**	**MY**
Government	0.000	0.500	1.000	3.000	3.000	3.000
Support Contractor	1.000	1.000	1.000	7.000	9.000	11.000
IBM Support	0.000	1.000	6.000	12.000	9.000	3.000
TOTAL	1.000	2.500	8.000	22.000	21.000	17.000

Table 3. PDSS Labor Estimate
From [Ref. 7, p. 71]

Cost Category	**FY93**	**FY94**	**FY95**	**FY96**	**FY97**	**TOTAL**
	$M	$M	$M	$M	$M	$M
Personnel Total	0.296	0.416	2.832	3.002	1.922	8.468
SEE Total	1.530	1.060	1.035	0.210	0.170	4.005
Total	1.826	1.476	3.867	3.212	2.092	12.473

Table 4. CECOM PDSS Funding Request
From [Ref. 45]

3. PDSS Process Action Team

Apparently unsure of the PDSS plan called for in the CRMP, the SOA Product Manager chartered a Process Action Team (PAT) in January 1993 to evaluate the various PDSS options available to the program, and to make a recommendation as to how to conduct PDSS "...in a manner that is cost effective and affordable." [Ref. 8, p. 2]

The PAT consisted of representatives from: the Product Manager's Office; the user; the contractors (IBM, Sikorsky, and Boeing); CECOM; and the Aviation and Troop Command (ATCOM). It initially considered nine PDSS options. As the PAT met over the period from January to May 1993, the field of PDSS options was narrowed to these four alternatives: [Refs. 8, 45]

- Alternative One - CECOM manages PDSS with its own SEE and contracting for IBM support as required.
- Alternative Two - The SOA PMO manages PDSS through the airframe contractors which would subcontract the effort to IBM.
- Alternative Three - The U.S. Air Force manages PDSS in manner similar to Alternative One.
- Alternative Four - The SOA PMO, and then the Technology Applications Program Office (TAPO) manages PDSS directly with IBM.

These four options were then compared in a cost benefit analysis which was documented in an Economic Analysis report. [Ref. 8] The analysis revealed that while Alternative Four was the second most expensive option, considering a 20 year discounted life cycle cost estimate, the benefits of this alternative outweighed the cost. The benefits considered for each alternative included: [Ref. 8, pp. 13 - 14]

- Maintenance Experience - The amount of maintenance experience in the organization. The maintenance task differs from that of development, and prior experience in maintenance is a benefit.
- SOA Experience - SOA experience consists of "domain" knowledge of the SOA software. Prior or current SOA experience is a benefit.
- No Transition - Transition is a description identifying the shift from the original software developer/maintainer. A loss of knowledge and familiarity, as well as

domain expertise, is to be expected during this shift. A lack of a transition is a benefit.

- Short Response Time - The time required for a software change to be implemented once the requirements have been determined. This is greatly impacted by the number of organizations involved. A short response time is a benefit.
- Facility - Facility is a measure of the availability of assets used to maintain the SOA software. A shared facility will impose a cost to timely and accurate software maintenance activities. A dedicated facility is a benefit.
- Translation of Requirements - The translation of requirements through multiple layers of communication is a concern. Fewer organizational layers is a benefit.
- Aircraft Knowledge - This attribute captures the amount of knowledge of the total aircraft system. This includes experience with the interaction and dependencies involved with the airframe and components, propulsion, pilot, and flight envelopes. Configuration management is also considered in this attribute.
- Other - These factors included the number of contracts involved, and the political considerations and possible lack of priority if the effort were performed by another service.

As a result of the PAT's recommendation, and the cost benefit analysis, the Product Manager elected to contract with the software developer, LORAL for the first three years of PDSS. The PM viewed this alternative as the "Best Value" approach to PDSS for the program. [Ref. 38] While this was not, at the time, viewed as the permanent direction for support of the software over the entire system life cycle, [Ref. 46] the future direction for PDSS beyond the first three years was not addressed by the PAT. On 25 July 1994, LORAL was awarded a one year firm-fixed-price contract, with two, one year options for PDSS of the Special Operations Aircraft. This contract, awarded initially as an undefinitized contract action, the base year of which purchased IAP Release 12.0, was definitized on 25 April 1995 at a price of $4.45 million, exactly the amount available for the effort. [Ref. 47]

4. The Key PDSS Players

The reader will recall from Chapter II that software acquisition management involves the interaction the key players - the user, the buyer, and the software developer in the acquisition process. This section identifies the key players involved in the SOA PDSS acquisition.

a. Technology Applications Program Office (TAPO)

On 1 April 1995, with the Product Manager having fulfilled his charter, the Project Management Office, Special Operations Aircraft (PM SOA) was disbanded and Program Management (PM) responsibility for the aircraft was transitioned to TAPO. TAPO serves as the "Materiel Developer" for the SOA and the other aircraft assigned to the 160th Special Operations Aviation Regiment (Airborne) (SOAR). It is under operational control of the U.S. Army Special Operations Integration Command and located in St. Louis with the Army Aviation and Troop Command (ATCOM). ATCOM provides TAPO's matrix support. This organization is shown in Figure 15.

TAPO's mission is to support the 160th SOAR with the streamlined acquisition of required aviation mission equipment. This mission includes: [Ref. 48]

- The performance of research and development as required.
- The use rapid, streamlined acquisition procedures to procure NDI mission equipment, aircraft systems, and aircraft
- Management SOF peculiar aircraft modifications, airworthiness release procedures, and configuration control.
- Providing for logistics sustainment of these aircraft systems.
- To be responsive to the needs of the user.
- When possible and practicable, transition SOF peculiar equipment to the conventional Army aircraft fleet.

TAPO is headed by a lieutenant colonel program manager. However, the day to day program management responsibility for the SOA is with one of the office's project officers. At present the SOA Project Officer (henceforth referred to as the PM), is an

Army aviator with experience as a SOA test pilot. In addition to the SOA PDSS effort, he is also responsible for other projects. Chief among them is responsibility for the ongoing effort to develop and qualify the terrain following (TF) capability of the MMR on the aircraft.

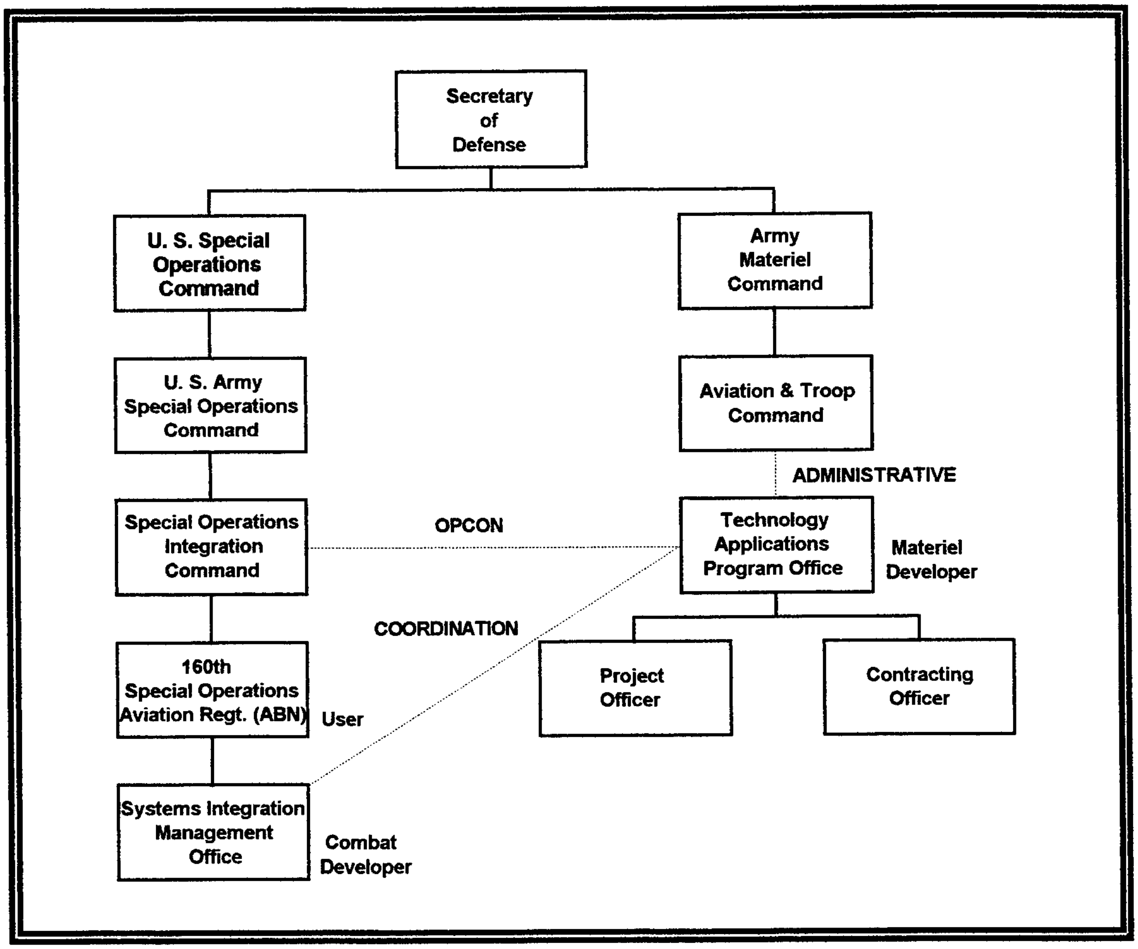

Figure 15. Special Operations Aircraft Organizations After [Ref. 48]

TAPO is sparsely manned, and lacks organic systems or software engineers to assist the PM in the management of PDSS. However, a software engineer has been hired on a support contract through the ATCOM Directorate for Lifecycle Software Engineering (LCSE). He assists the PM in the preparation of statements of work and technical reviews of LORAL's proposals. Additionally, the PM has provided some funds to the Software Engineering Directorate at the U.S. Missile Command (MICOM), Redstone Arsenal, Alabama, and has been assisted for several months by three of MICOM's software engineers. The primary purpose of their assistance is to help the PM to be a "smart buyer" in the PDSS contract with LORAL.

There are numerous PDSS issues which concern the PM. These issues include: [Ref. 49]

- The apparent high cost of $4.45 million for the base year of PDSS which delivered IAP Source Release 12.0;
- Identifying the content/functionality of the changes to be incorporated in the next software release;
- Management of the future growth and configuration of the IAS;
- Developing a capability to conduct impact assessments and cost estimates for software changes; and
- Negotiating with LORAL to obtain all of the desired functionality for a given software release, within the allocated budget.

The bulk of the PM's time spent on the PDSS program is spent in the management of the ongoing issues and problems, and fighting the current "brush fires." However, in spite of an environment which allows little time for planning, the PM's vision in regard to the SOA PDSS program is: [Ref. 50]

- To be a "smart buyer" who can effectively manage the current PDSS contract to ensure the efficient, cost effective delivery of software support services.
- To identify the future direction for PDSS of the SOA.

b. The Systems Integration Management Office (SIMO)

SIMO represents the user, the 160th SOAR and serves as the combat developer in the acquisition management process. In this role, it maintains a data base of the Government Trouble Reports (GTRs) written against the SOA software. From this data base, SIMO selects GTRs to define the PDSS requirement and the content of the yearly software release. As one would reasonably expect, SIMO's primary goal in regard to PDSS is maintaining a software product that continues to meet the needs of the user. However, another typical user goal, which was discussed in Chapter II, is the continued enhancement of the SOA system's capabilities, and this at a minimum cost. Based on previous experience with the other user aircraft systems, it is reasonable to expect that the user will continue to request enhancement of the SOA IAS. While generally pleased with the quality and function of the software, SIMO representatives have mentioned "going competitive" after the current contract. These comments were based primarily on the perceived high cost of PDSS.

c. LORAL Federal Systems Company

LORAL, as the former IBM Federal Systems Division, was the Army directed subcontractor which developed the SOA IAS and its software. It is the lead contractor in the development, integration and qualification of the TF capability of the MMR. In addition, as the PDSS contractor, the company is adapting to a relatively new relationship, working directly with the Army as the lead contractor on the SOA program. With a well staffed program office and organizations which support systems and software engineering, the company prides itself on its performance as the SOA systems integrator, and on the quality of the SOA software. As the system integrator, LORAL's goals are: [Ref. 39, pp. 15-16]

- A SOA system that is safe to fly.
- System configuration control to assure safe/reliable operation.
- Being a cost effective solution to the SOA program's needs with:
  - Facilities and trained personnel in place.
  - Extensive use of modeling and simulation for efficient system and software development.
  - Cost effective and efficient management of subcontractors.

LORAL's overall goal is a satisfied customer. It plans on staying in this business area and believes that it has the skills and resources to provide a "complete solution" for the SOA program. LORAL believes that the key to the future of this complex program is a closer, "team" relationship among the user, the PM, and LORAL. It believes that this type of relationship will foster a deeper understanding of both the program's needs, and the potential solutions. [Refs. 39, 51]

5. The PDSS Acquisition Management Process

The reader may recall from Chapter II that software acquisition management is "...the process of acquiring custom software, usually by contract from some third party", and that it includes activities such as: planning, contracting, budgeting, evaluating and managing performance, and providing for the future support of the system. [Ref. 11, p. 50] The sections below address the planning, budgeting, and the contracting aspects of the SOA PDSS acquisition. The current process of evaluating and managing performance will be addressed under Subsection 6 - The PDSS Project Management Process. As for providing for the future support of the system, this has yet to be addressed in any detail.

a. Planning

The primary planning tool utilized under the current PDSS program is the statement of work (SOW). It defines the scope of the software effort to be accomplished by LORAL for the particular year of the contract. [Refs. 52, 53] The SOW is developed by the PM in conjunction with his one matrix software engineer and input from the user pilots and SIMO. In both the base and first option year, the Government had to revise the SOW during the proposal/negotiation process. This was done so that the scope of work could be accomplished within budget. The SOW establishes two separate processes for PDSS during the year of support. The first process is the development of the new source software release for the IAP. The second is a process to handle the correction of priority one and two STRs which may occur during the year. These are software problems which either prevent the aircraft from performing its mission, or which severely degrade mission performance.

The key tools used in the first process, defining the content of year's new IAP software release, are the databases which are maintained to track software trouble reports. The first database, the Government Trouble Report (GTR) database is maintained by

SIMO and documents trouble reports. These are problems with the software reported by the user pilots. The second database, the Common Database (CDB) is maintained by LORAL and documents all STRs, both GTRs and those found during LORAL's software development and testing effort. These databases provide a common framework with which to define what new functionality will be added to the current fielded baseline IAP software. For example, the SOW for the base year of the contract, [Ref. 52] which yielded IAP Source Release 12.0 specified those specific GTR numbers which were to be implemented into the new source release.

The second process is the correction of any priority one or two STRs. This is accomplished as a separate effort, "over and above" the base effort conducted to produce the IAP source release. Under this process, LORAL begins investigation of the reported STR within 24 hours, and recommends corrective action to the PM within five calendar days. The recommendation includes an estimate of the cost and schedule to implement a correction, and an estimate of the impact the correction will have on the system. If approved, LORAL takes action and implements the correction.

While these are the two main efforts included in the SOW, it also addresses the following topics:

- Software documentation.
- Support of flight testing of the new source release.
- Program management reviews.

b. Budgeting for PDSS

The PDSS effort is funded primarily with Major Force Program (MFP) 11, Operations and Maintenance (O&M) funds. Funds are allocated to TAPO for PDSS through the U.S. Army Special Operations Command. Currently the SOA PDSS program is operating on funds which were budgeted several years ago by PM SOA. Future years are budgeted by SIMO in accordance with the Planning, Programming and Budgeting System (PPBS). Under this process, SIMO is currently working on the budget estimate for the Program Objectives Memorandum (POM) years 1998 through 2003. In developing budget projections, SIMO considers two aspects of PDSS. First it considers basic software support, funded with O&M funds. It projects this amount based on what has been spent to date, plus an inflation factor. Second, it considers projected system enhancements, such as integration of new avionics components. For these efforts, it budgets research and development funds (RDT&E), procurement funds for purchase of new hardware components, and O&M funds for software integration. Estimates for these future efforts are essentially "best guesses." [Ref. 54]

c. The PDSS Contract

The current PDSS contract was awarded on 25 July 1994 as a firm-fixed-price contract, with two option periods. The period of performance for both the base and the options is one year. A firm-fixed-price contract was selected essentially to limit the Government's risk in the correction of any software errors discovered during TF development of the MMR. At that time it was not known how large an effort it would be. The fear of this risk was motivated at least in part from events which occurred during the software development, when there were disagreements among the ACs, IBM, and the Government over who had the responsibility to correct software anomalies which were considered "out of scope" under the firm-fixed-price development and production contracts. [Ref. 55] Under the first year of the contract, the over and above line items for correction of priority one and two STRs, and other software efforts were also firm-fixed-price. In the first option year, these line items have been changed to a time and materials (T&M) basis. An example of this is the effort to integrate a new Digital Map processor into the IAS. This effort, estimated to cost $2 million will be accomplished on a T&M basis.

6. The PDSS Project Management Process

Project management is the process by which the software development is monitored and controlled. Within the context of the SOA PDSS case, the areas of management reviews and metrics will be discussed.

a. Reviews

Together, the PDSS contract SOW [Refs. 52, 53] and LORAL's Software Development Plan [Ref. 40] establish a program of management reviews. These reviews and their purpose are discussed below:

- Program Reviews - Program Reviews are held on an as needed basis for the purpose of discussing resolution of problems, methodology, documentation, and milestones.
- Technical Interchange Meetings (TIM) - TIMs are held periodically during the software development cycle to discuss the implementation, the possible rework, or the deletion of the GTRs which were specified to be included in the source software release.
- Flight Readiness Review (FRR) - The Flight Readiness Review is conducted prior to flight testing of the software release to review the Flight Test Plan. Following flight testing of the software, a Flight Test Report is generated by LORAL. Acceptance of this report documents the Government's acceptance of the software.
- User Lab Tests - In addition to the reviews formally required in the SOW, User Lab Tests have been conducted periodically during software development to obtain user input on the way in which GTRs are being implemented. [Ref. 56]

b. Metrics

The only metrics called for in the PDSS SOW is maintenance of the Common Database. LORAL does collect and periodically provides metrics concerning the growth in the size of the IAP, and processor and memory utilization, but there is no formal metrics program required in the SOW.

7. The PDSS Systems/Software Engineering Process and Tools

This section describes the systems and software engineering processes, and the software engineering tools in use on the SOA PDSS program. Software engineering is the "...technological and managerial discipline concerned with systematic production and maintenance of software products..." [Ref. 3, p. 5-1] Systems engineering is defined as:

> ...the management function which controls the total system development for the purposes of achieving an optimum balance of all system elements. It is a process which transforms an operational need into a description of system parameters and integrates those parameters to optimize the overall system effectiveness. [Ref. 57, p. 1-2]

a. The Systems/Software Engineering Process

Support of the SOA IAS is more than simply an exercise in computer programming. As a highly complex, integrated system consisting of both hardware and software, systems engineering is a critical function in the management of the system as it grows and evolves through the system life-cycle. Systems engineers are responsible for assessing the impact of any changes to be made in the software and /or IAS hardware, for making the required changes to the Software Requirements Specification (SRS), and for conducting the system level integration and testing. [Ref. 40, p. 103] Other systems engineering activities are shown in Figure 16. The software engineering process essentially mirrors the process described in DoD Standard 2167A. While LORAL did begin to incorporate some prototyping in the development of IAP Release 12.0, the process can be characterized predominantly as a waterfall process, similar to that shown in Figure 5.

b. Software Engineering Tools

The SOA IAS was developed and is maintained on multiple software toolsets. Recall from Figure 14, that of the nine CSCIs in the IAS, LORAL developed and maintains only the IAP and the SROMP at its facility; the other CSCIs are maintained by LORAL through subcontracts. The IAP, the heart of the system was developed in

JOVIAL. This is an older third generation Higher Order Language (HOL). The toolset on which it was developed is also old technology, dating from the 1980s and cannot support many of the recently developed capabilities in software engineering. LORAL's major IAS subcontractor is Allied-Signal Aerospace Company, Flight Systems Division. The Allied CSCIs include the DP, RTU, and the MDG. Figure 14 also shows that these CSCIs were developed in several languages, including JOVIAL, Ada, and assembly language. Assembly languages are second generation programming languages which differ for each type of computer processor. The CDU CSCI was developed by Smith Industries Aerospace in assembly language. Thus, the IAS development and maintenance is conducted on a distributed architecture. This involves activities at three contractors, with six separate languages and the tools to support them. [Ref. 7]

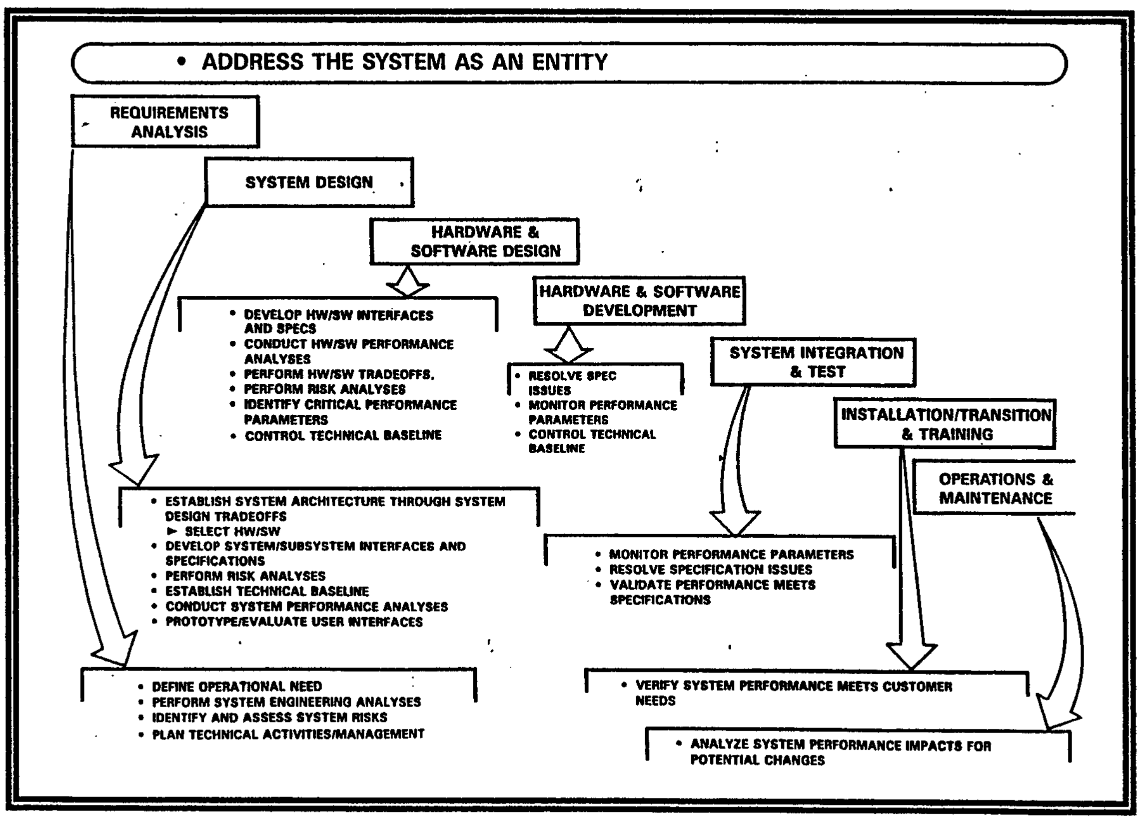

Figure 16. Systems Engineering Activities
From [Ref. 39]

8. PDSS Year One

The base year of the PDSS contract was awarded as an undefinitized contract action on 25 July 1994. It was negotiated by the administrative contracting officer, and definitized on 25 April 1995. The primary effort under the contract was to develop and deliver IAP Release 12.0. The content was defined as a set of approximately 40 GTRs, most of which would be classified as user enhancements. As a risk limitation measure under the firm-fixed-price contract, LORAL required a limit of 1,675 source lines of code (SLOC) to be placed in the SOW. However, LORAL implemented the functionality required by the GTRs within the firm-fixed-price of $4.45 million, but far exceeded the SLOC estimate of 1,675, delivering approximately 5,400 SLOC.

9. PDSS Year Two

The SOW for the first option year of PDSS calls for the development and delivery of IAP Release 13.0. The major task for this source release will be to implement the software "patches" which were developed during the development and qualification of the TF capability of the MMR. Patches are essentially separate programs which run with the IAP, but are not compiled in the IAP source code. The software changes for Release 13.0 have already been developed in these patches, but will have to be implemented in the IAP source code, and then integrated and tested. In addition to this effort, on a time and materials basis, a new LRU called the Digital Map Processor will be integrated into the IAS, replacing the current Map Display Generator. This effort is estimated to cost $2 million and is outside the base effort.

Negotiations for the first option year took place in late August and early September 1995. Because the effort was funded with expiring O&M funds, there was great pressure to negotiate the price for the option quickly. At the first round of negotiations, the PM's supporting software engineer, and the MICOM software engineers presented the Government's technical position on the effort. The technical position was based on the engineers' analysis of LORAL's proposal. The Government's technical position on labor hours, and its cost position were approximately half the amount proposed by LORAL. From LORAL's perspective, the Government's position was so dramatically different from its own that it represented a different scope of work than that specified in the SOW. LORAL's program manager expressed the following concerns about the Government's position and the PDSS effort in general: [Ref. 58]

- The engineering hours in the Government position would require LORAL to expedite delivery of Release 13.0 in December 1995. SOA software development personnel would then have to be moved to other projects, leaving the program without dedicated support until additional over and above efforts were funded.
- The Government position calls for the elimination of software verification testing. This activity ensures that changes in the CSCIs are traced back to the software requirements specification.
- The exit criteria for flight testing, which documents acceptance of the software is not clear. LORAL faces what it sees as an unbounded risk under the firm-fixed-price contract for investigation of all STRs found during flight testing of Release 13.0. In past flight tests, most of the STRs reported by the pilots have not been LORAL's responsibility.

Following this initial meeting, LORAL negotiators returned to the plant to develop a revised proposal. Negotiations were settled approximately a week later at a firm-fixed-price of $2.8 million. This price provides what LORAL called a "framework" - which essentially keeps the SOA development staff committed to the program for the entire year of support.

D. CHAPTER SUMMARY

The Special Operations Aircraft program began in response to the tragic circumstances which caused the failure of the Iranian hostage rescue mission in April 1980. As the White House Statement shown at the beginning of the chapter indicated, the mission was terminated "...because of equipment failure." Today, the MH-60K and MH-47E Special Operations Aircraft are equipped with a Mission Equipment Package (MEP) which allows the aircraft to operate reliably under the demanding conditions encountered by the aviators of the 160th Special Operations Aviation Regiment. The heart of the MEP is the software intensive SOA Integrated Avionics Subsystem (IAS) which reduces pilot workload in mission planning, navigation, and communication. As the IAS continues to evolve in response to changing user requirements, the effective management of the software system through PDSS is essential to ensure that there is never a repeat of "Desert One."

IV. ANALYSIS AND LESSONS LEARNED

A. INTRODUCTION

This chapter provides the analysis of the SOA PDSS case described in Chapter III. The objective of this analysis is to identify the significant management issues present in the SOA PDSS case and to present the lessons learned from analysis of these issues. The reader will recall from Chapter II, that software acquisition management involves the interaction of the buyer, the seller, and the user in three overlapping processes as shown below in Figure 17.

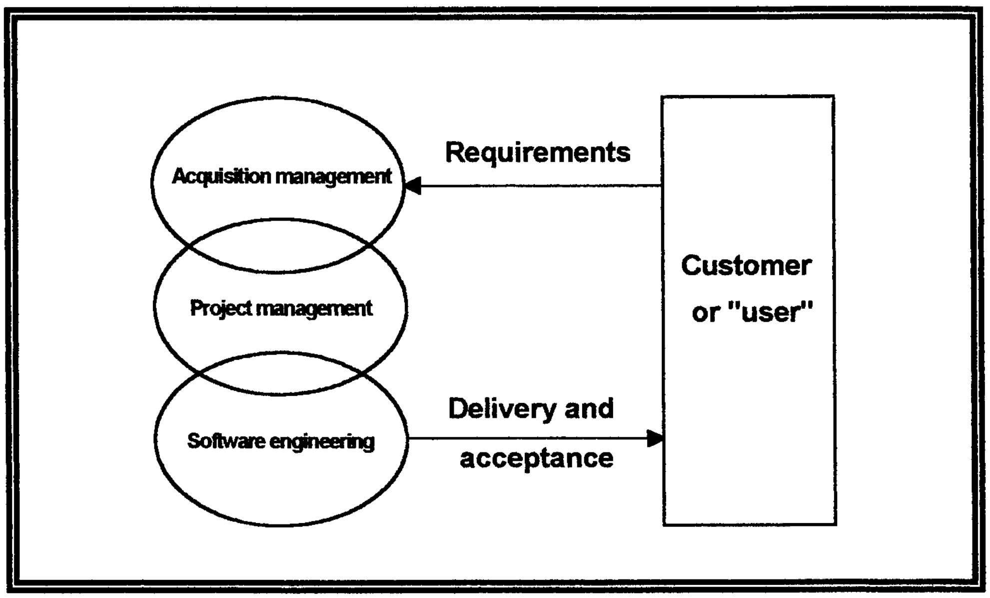

Figure 17. The Software Acquisition Management Relationships From [Ref. 11, p. 51]

The acquisition management process is primarily the responsibility of the PM and his supporting contracting officer. The software engineering process is primarily the responsibility of the software development contractor, and both the PM and the contractor share responsibility for the project management process. This model provides a convenient framework within which to identify and analyze the significant issues which impact the management of the SOA PDSS program.

B. ACQUISITION MANAGEMENT ISSUES

The acquisition management process includes those activities undertaken by the PM to plan, budget, and contract for the software acquisition. The significant acquisition management issues identified in the SOA PDSS case are in the areas of cost, program plans, the buyer-seller relationship, and the contract type. These issues are analyzed in the following sections.

1. Cost

The most significant underlying issue involved in the SOA PDSS acquisition, from the Government's perspective, is the cost of the PDSS contract. The cost issue is characterized by a Government perspective that the LORAL PDSS contract costs too much and is inefficient. This perception is the cause of considerable friction between the Government and the contractor. This friction in turn has fostered a harmful short-term perspective, and mistrust on the part of the Army (both the PM and the user) in regard to the buyer-seller relationship.

The cost of the PDSS contract is presently viewed as too high by the PM. This research however, leads one to ask: By what standard is the PDSS cost too high? The PDSS contract is essentially a new effort for LORAL. As such, there is no valid basis for comparison of the PDSS cost to LORAL's historical development cost data. The Defense Contract Audit Agency has confirmed this. In its audit of LORAL's base year PDSS proposal, it stated that because the software maintenance effort is essentially new, historical analysis of the proposed labor cost could not be accomplished. [Ref. 59]

One question which could be asked: Is the cost too high in comparison with what other aviation programs pay for PDSS? Since the SOA is a unique system, it is difficult to compare it with other aviation systems. For example, the Navy's LAMPS helicopter program, also supported by LORAL, spends about two million dollars per year for PDSS. The SOA's base year PDSS cost was $4.45 million. [Ref. 60] One cannot simply compare costs. The software systems are different in purpose and scope.

A second question: Is the cost too high in relation to the Government's own estimate of the effort required to implement the requested changes? When one examines this issue it becomes clear that the Government does not currently have the system knowledge, expertise, or capability to perform its own detailed analysis of what changes to the IAS software should cost. The PM's matrix support software engineer, and the MICOM software engineers were effective in negotiating a lower price for the first option year of PDSS. However, this success was based upon cutting the scope of work, and on decrements to LORAL's proposal. It was not based on an independent cost estimate of the effort specified in the SOW. Estimates of this nature require both software engineering expertise, and detailed system specific expertise. While the PM's supporting software engineers are experts, the detailed system specific expertise currently resides only in LORAL.

Another question: Is the cost too high in relation to what was budgeted for PDSS? The answer - perhaps. The budget for a given year's PDSS effort is prepared at least two years in advance of the actual budget year. At such a time, the extent of the changes to be made to the system two years in advance were not forecast in detail. As a result, it is not surprising that the budget does not support the purchase of all of the new functionality now desired.

From a Government perspective, the primary complaint is with regard to the categories of cost which are viewed as "non-value added" as compared to the direct effort of software support. These include categories such as proposal preparation and program management, which accounted for 30 percent of the base year cost. The distribution of the actual base year PDSS costs, and the proposed option year 1 costs are shown in Figure 18. While generally dissatisfied with the proportions of these cost categories, the Army must admit that at least part of the responsibility for such costs lies with the Government.

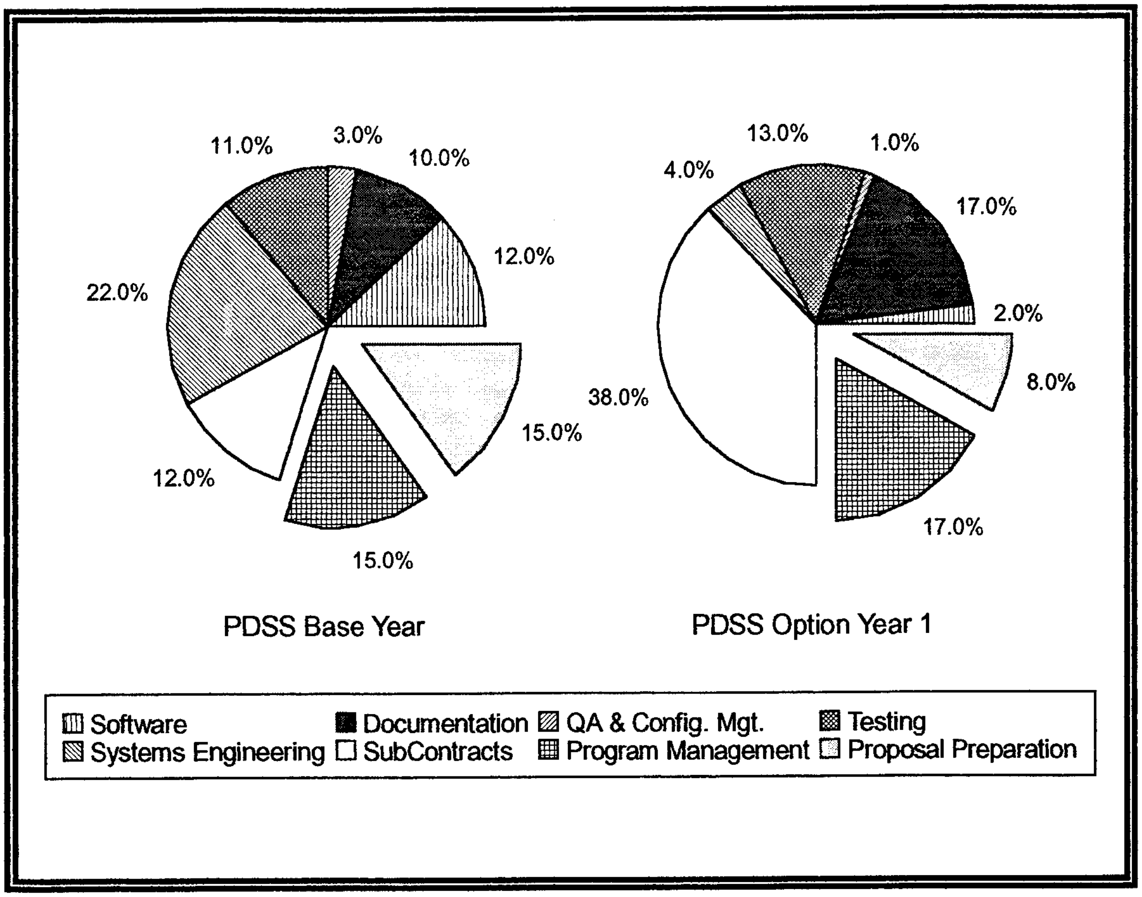

Figure 18. PDSS Cost Distribution
From [Ref. 61]

For example, proposal preparation costs, at eight to 15 percent of costs, are partially impacted by the several iterations of SOWs to which the contractor has typically had to react before the Government finally defines the scope of the PDSS effort. While program management costs do appear to be high, there has been no direct effort on the part of the Government, other than during negotiations to reduce it. From LORAL's perspective, this category of cost is required and simply reflects the level of management necessary to run the SOA PDSS program. This may be an area which could be addressed in an incentive type contract. A closer, more cooperative working relationship with LORAL could also significantly impact this area.

In addition to these areas, which the Government views as non-value added, the cost of PDSS is also necessarily a function of the scope of the effort specified in the SOW.

Three of the major tasks specified in the SOW are the IAP software release, software documentation, and the DP source release, a subcontractor effort.

Early in the development cycle, during the PDSS base year, the SOA PM Technical Division Director performed an informal analysis of the content of IAP release 12.0. This analysis revealed that approximately 87% of the additional function added to the software was in the form of user requested enhancements. [Ref. 62] The remainder was in the form of corrective actions. Such system improvements add real value to the system.

A major factor that impacts the cost of PDSS is the amount of documentation which the Government buys. Documentation is defined as the "paperware" which, "...provides the means to describe the product and the engineering which lead to its development." [Ref. 19, p. 24] Figure 18 shows that documentation accounted for ten percent of the actual cost of the base year contract. For the first option year, 17 percent of the contract price will go toward documentation. The cost of documentation is impacted by the fact that the future direction for PDSS after the current contract is still in question (i.e., if the Government plans to take over PDSS, or transition PDSS to another contractor, detailed documentation will be required). If a PDSS strategy is confirmed, then the type and amount of documentation could be tailored specifically to support the selected strategy. The third area of cost directly resulting from the scope of work specified in the SOW is the DP source release. This cost is shown in Figure 18 as subcontract cost.

Thus, the total cost of PDSS is a function of not only the software products developed, but also of the joint Government-LORAL process for arriving at that product. This view is confirmed by researchers Haimes and Chittister. [Ref. 63] In research concerning the management of cost and schedule risk in software development, they concluded that the sources of risk in these areas can be attributed to both the buyer and the seller.

In terms of the contractor, several of the factors which contribute to cost and schedule risk are: [Ref. 63, pp. 133-134]

- Organizational maturity level.
- The process and procedures followed in the assessment of the project's cost and schedule.
- The level of honesty exhibited by management in communicating the real cost and schedule to the customer (and of course vice versa).
- The level of experience and expertise of the staff engineers in software engineering, in general, and in the application domain in particular.
- The level of experience and expertise of the management team in software engineering.
- Financial and competitive considerations.

In terms of the buyer, Haimes and Chittister found that the following factors influence cost and schedule risk: [Ref. 63, p. 134]

- The process and procedures followed in the assessment of the project's cost and schedule.
- The level of specificity at which the system and software requirements are detailed.
- The number of changes and modifications requested by the customer during the software development process which are often not harmonious with earlier specification requirements.
- The commitment of the customer's project management to monitor and oversee the software development process.
- The level of honesty exhibited by management in communicating the real cost to the "real client," the U.S. Congress.

The true issue in regard to cost is not its relative magnitude. Rather, it is how it is jointly managed by the Government and LORAL. Both parties have an interest in maintaining an affordable PDSS program. The Government's interest is the continuity of affordable, quality software maintenance support. LORAL's interest is in satisfying the SOA customers and keeping the SOA business. The issue of cost is critical because it influences, or is influenced by nearly all of the other issues in the SOA PDSS program.

2. Program Plans

Development of program strategies and plans is one of a PM's major acquisition management functions. Program plans establish the overall framework for the management of the program. Currently, the only acquisition planning mechanism in use in the SOA PDSS acquisition is the SOW for the current year's PDSS effort. This is a significant issue because lack of a longer range plan fosters a harmful short term management perspective. The key program plan in effective management of PDSS is the Computer Resources Life-Cycle Management Plan (CRLCMP). [Ref. 13, p. 6-D-2] The CRLCMP serves to document the PDSS concept or strategy, and the detailed approach to implement that concept. The existing SOA version of the CRLCMP, the CRMP was last updated in June 1992. It reflects the former CECOM support concept discussed in Chapter III. This concept called for the purchase of a complete SEE with which to maintain all of the IAS CSCIs. Transition from LORAL to CECOM was to have been accomplished over several years, during which LORAL's participation would be gradually phased out.

The PM fully recognizes the need for a long range strategy for PDSS. However, as essentially a one man operation, he has little time to spend, and few support resources to assist in the development of an acquisition strategy. The importance of acquisition strategy and planning is firmly supported in policy. The term "acquisition strategy" refers primarily to the formal strategy required by the DoDI 5000.2 for a major acquisition program. However, a related concept which is more appropriate in this case is that of acquisition planning.

In reference to acquisition planning, FAR Part 7.101 states that:

> "Acquisition planning" means the process by which the efforts of all personnel responsible for an acquisition are coordinated and integrated through a comprehensive plan for fulfilling the agency need in a timely manner and at a reasonable cost. It includes developing the overall strategy for managing the acquisition. [Ref. 64, Part 7.101]

This issue seriously impacts the SOA PDSS program because failure to identify the PDSS program's future direction impacts the decisions presently being made. An example of this is the decision of how much, and what type of documentation to buy. Currently, the PM and the user appear to be contemplating two different directions for the future of the PDSS program after this current contract is completed. The user, although satisfied with the quality and the performance of the software, is very dissatisfied with its cost. As a result, the user wants to procure PDSS services competitively. In contrast, it appears that the PM recognizes that transition to another source would be a risky venture. The PM would prefer to continue working with LORAL, at least until the Government reduces its risk through the development of a base of expertise in the SOA IAS software, and in general learns to manage software acquisition.

A competitive procurement conducted solely in the "hope" of finding a cheaper source would be extremely risky, and may be "penny wise and pound foolish." There are two sources of risk in this area. First is the risk in maintaining working relationships with LORAL's IAS subcontractors, as well as the relationship with LORAL on the SOA Life Cycle Contractor Support (LCCS) Contract. The reader will recall from Chapter III that seven of the nine IAS CSCIs were developed and are currently maintained by subcontractors. The Government has depended on LORAL to manage these subcontractors in maintaining the IAS. How would changing sources impact these relationships and the program's ability to maintain responsive support from these subcontractors? Additionally, LORAL is a major subcontractor to BSAS. As such it is responsible for the depot level maintenance of the AP102A MP hardware. How would changing sources impact operations on that contract?

The second source of risk in changing PDSS sources concerns the complexity of the IAS. Because of the complexity of the SOA IAS, and the strategic importance of the user's mission, systems and software engineering expertise is critical in maintaining the system. This point has been illustrated in at least two other complex, software intensive DoD systems. The Navy's LAMPS program transitioned software support from LORAL to a Navy SSA. It has since elected to transition back to LORAL. The primary reason for this reversal was for the systems engineering expertise at LORAL. A second reason was that the cost of maintaining two software engineering environments (i.e., one at LORAL and one at the SSA) was prohibitive. These costs included: maintenance of dual sets of avionics components and laboratory hardware; and the development tools. [Ref. 60] The Army's PATRIOT Missile system underwent a similar transition. It also has since transitioned support back to the original software developer.

The PATRIOT's software engineering chief made this comment concerning the folly of pursuing the lowest bidder for software development:

> It is false economy in software development to go get the low bidder (Government or contractor) to do your software. In complex systems, if the low bidder doesn't have the requisite expertise or experience he will encounter major problems and costs will soar. In many areas, ten people who lack system specific experience simply cannot do the critical job that one experienced person can do. Throwing cheap people at complex software jobs is not a viable answer. [Ref. 65]

The SOA PDSS process action team (PAT), charged with the original PDSS assessment included user representation. It also recognized the importance of system specific expertise. In considering which PDSS alternative to recommend to the PM SOA, it considered the following factors: [Ref. 8]

- Maintenance Experience - The amount of maintenance experience in the organization. The maintenance task differs from that of development, and prior experience in maintenance was considered a benefit.
- SOA Experience - SOA experience consists of "domain" knowledge of the SOA software. Prior or current SOA software experience was considered a benefit.
- No Transition - Transition is a description identifying the shift from the original software developer/maintainer. A loss of knowledge and familiarity as well as domain expertise is to be expected during this shift. Because of this, not transitioning was considered a benefit.
- Aircraft Knowledge - This attribute captures the amount of knowledge of the total aircraft system. This includes experience with the interaction and dependencies involved with the airframe and components, propulsion, pilot, and

flight envelopes. Configuration management of the aircraft system was also considered in this attribute.

If these factors were important in 1993, are they any less important now? Clearly, these factors are as important now as they were when the PAT considered the various PDSS options in 1993. It was these factors that lead to selection of LORAL to perform PDSS, despite the higher cost of that alternative.

While the PM lacks a written acquisition strategy or plan, this is but one dimension of the concept of strategy. Another dimension of strategy addressed by author Henry Mintzberg [Ref. 66, p. 13] is that strategy is also a pattern. In this sense, strategy is a ***pattern in a stream of actions***. Under this definition, strategy is not a plan, but rather is represented by *consistency in behavior, whether or not intended.* In the researchers view, there is a definite pattern of actions which indicate a PDSS strategy which centers on LORAL. This pattern indicates growing dependence on LORAL as the SOA systems integrator, and thus as the only firm (at this time) that can *realistically* accomplish the enhancements which the user requires during PDSS. The pattern of actions includes:

- LORAL's role as a key LCCS subcontractor.
- The selection of LORAL to accomplish the integration and qualification of the TF capability of the mulitmode radar.
- Departing from the original PDSS concept which involved a transition to Government support at CECOM, to a contract with LORAL.
- A continued stream of system enhancements implemented by LORAL. These include the integration of a new digital map processor, and the embedded GPS inertial. These enhancements further complicate the original system.
- Avoiding the risk reduction actions to prepare for transition to another source. Such actions might include a transition assessment, the use of an independent verification and validation contractor, and development of a transition plan.

While there is no doubt that another source could eventually take over the PDSS effort, one must ask, what would be the price (in terms of cost, quality, and responsiveness of support)? This is not to say that SOA must always be supported by LORAL, but rather that transition is risky and such risk must be confronted. There are unknown costs and risks involved in changing sources, and a hasty transition could potentially leave the SOA IAS without support. A transition to another source would require detailed planning, and a coordinated sequence of hand-offs as specified in the discussion of Pre-Deployment Software Support in Chapter II.

The second issue in regard to program plans is the planning of the evolution of the IAS. This includes establishing a link between the desired system changes and enhancements, and the budget. The program requirements, in the form of enhancements, (such as the Digital Map Processor integration) for the base and first option year had not been tied in any recognizable way to the PDSS budget. Since detailed Budget Estimate Submissions are sent into the Planning, Programming, and Budgeting System (PPBS) two years prior to a given budget year, [Ref. 67] PDSS requirements must be identified, estimated, and budgeted at least that far in advance. According to a SIMO representative, the PDSS budget is now being planned in accordance with the PPBS. SIMO is currently working on the POM for the years 1998 through 2003. However, without SOA knowledgeable system and software engineers working on these budget estimates, their accuracy is in question. Additionally, there is no single integrated plan which documents planned IAS enhancements and the corresponding budget. This is an issue because the PM cannot make long range plans without visibility into the future requirements and the proper budget to implement those requirements.

Another dimension of this issue is the type of funds used for PDSS. It is understood that maintenance of the current software baseline is accomplished with O&M funds. However, the addition of new functions or hardware which create a new baseline should be implemented with the proper combination of RDT&E and procurement funds. The integration of the new digital map processor can serve as an example. Both the purchase of any new avionics components and their integration/software development should be accomplished with RDT&E and/or procurement funds. Use of O&M funds for integration of new hardware, distorts the true cost of supporting the existing software baseline.

In summary, there are two major issues in regard to planning on the SOA PDSS program. First is the lack of a clear acquisition strategy for PDSS after the current contract. This lack of clear direction fosters a harmful short term management perspective. The second major issue is the planning of the evolution of the IAS. This includes the budget required to implement system enhancements. This is an issue because the PM cannot make long range plans without visibility into the future requirements and the proper budget to implement those requirements.

3. The Buyer-Seller Relationship

One of the most serious issues in regard to the SOA PDSS acquisition is the level of trust and cooperation in the buyer-seller relationship. While the Government and LORAL are dependent on each other in this relationship, the present level of trust and cooperation does not reflect this level of mutual dependence. As a result, issues such as cost, (which could potentially be addressed through a closer working relationship), are left unsolved. One indication of the level of mistrust is the selection of a firm-fixed-price contract. It places all of the cost risk on LORAL for effort that is largely developmental. Much of this mistrust on the part of the Government, specifically among the user community, is left over from the SOA development and production programs. During those programs, there were disputes over exactly what functionality the PIDS required, and which of the contractor's, (the ACs or LORAL) would be responsible for making the required changes to meet the user's desires. All of this was the direct result of the fact that all contracts in the program were firm-fixed-price, and did not reflect the obvious level of development risk to all the parties involved.

The same situation is present to a degree in the PDSS contract. The situation under the current firm-fixed-price contract finds the Government attempting to get all the additional software functionality it can within its budget. And LORAL attempting to protect itself from the risks involved in proposing firm-fixed-prices for development of new software functions. This situation is almost certain to create an adversarial relationship. This puts the parties further apart than the required "arms length relationship."

The benefit of closer working relationships between the Government and its contractors is supported both in theory and in policy. In examining the theoretical approaches to the buyer-seller relationship in Defense contracting, Lieutenant Colonel Carl R. Templin [Ref. 68] examined the degree of "coupling" between the buyer and the seller. Based on the research of Landeros and Monczka [Ref. 69], the theory examines three degrees of coupling in the buyer-seller relationship. The degree of coupling present in the relationship is defined by attributes such as:

- The number of suppliers available to the seller;
- The degree of commitment or "alliance" between the parties;
- How the parties resolve disputes;
- The flow of information between the parties, and;
- How the parties adjust to a changing market.

These factors and the degree of coupling in the relationship are depicted graphically in Figure 19. The model in Figure 19 shows the degree of coupling between the parties may vary on a continuum, from loosely coupled to fully coupled. On the left of the continuum, a loosely coupled relationship is one in which the independence of the parties is maintained through open market bargaining. In DoD this is characterized by a contractual relationship based on sealed bidding, where price is the sole source selection factor, and the contract type is firm-fixed-price. On the right side of the continuum is a fully coupled, or vertically integrated relationship. A DoD example of this is an organic Government software support activity.

Between these two extremes are tightly coupled relationships. These relationships are based on cooperative buyer-seller relationships which are designed to achieve mutually beneficial long term strategic goals. These goals may include cost reduction, better performance, and improved quality and reliability. [Ref. 68, p. 124] This type of relationship is characterized by a considerable degree of interdependency between the parties. Because of this degree of interdependency, there is also a high degree of cooperation. Comparing this theoretical model to the relationship between the Government and LORAL, it appears that there is a mismatching of the degree of coupling and the attributes which define the situation.

Attribute:	Loosely Coupled (Market Bargaining)	Tightly Coupled (Cooperative Relationship)	Fully Coupled (Vertical Integration)
Supply Pool	Numerous Suppliers	↔	One Supplier
Alliance	Credible Threat	↔	Credible Commitment
Dispute Resolution	Unyielding Negotiations	↔	Managerial Tradeoffs
Information Exchange	Minimal	↔	Great
Marketplace Adjustment	Separate	↔	Joint

Figure 19. Buyer-Seller System Coupling
From [Ref. 68]

In this situation, the degree of interdependence between the Government and LORAL is high. LORAL is dependent on the SOA business to keep personnel employed. The Government is dependent on LORAL's SOA system knowledge and expertise. This interdependence indicates that a tightly coupled relationship, in the center of the continuum is appropriate in this situation. However, several of the attributes of the relationship indicate that the parties believe that this a loosely coupled relationship, able to be controlled by market bargaining.

This is demonstrated by several of the attributes which define the relationship. First the Government believes that there are numerous suppliers available which could compete for the PDSS contract and therefore drive down the cost. In fact, at this time in the program, the pool of ***realistically*** qualified suppliers available to the Government is one - LORAL. Secondly, the level of commitment or alliance is low. The Government's use of the "going competitive" threat indicates a low level of commitment in the relationship. Thirdly, the level of information exchange, especially in the area of software process metrics is low. Application of this model indicates that the relationship would be most effective with increased cooperation. But instead, it is presently managed as a loosely coupled relationship.

The value of cooperative buyer-seller relationships is also supported in policy. The recently approved Statement of Guiding Principles for the Federal Acquisition System [Ref. 70] seeks an environment which fosters cooperative relationships between the Government and its contractors, consistent with the overriding responsibility to the taxpayer.

Thus, the level of mistrust and the lack of cooperation in the buyer-seller relationship is a major issue. It prevents joint problem solving of issues which impact both sides of the relationship.

4. Contract Type

The single most critical document in the acquisition of weapon systems software, or in this case the acquisition of software support, is the contract itself. The contract defines the legal relationship and the obligations which exists between the buyer and seller. Selection of the contract type is generally a matter for negotiation and requires the exercise of sound judgment. The negotiation of the contract type is closely related to negotiation of the price and the two elements should be considered together. The objective is to negotiate a contract type and price which reasonably allocates risk between the contractor and the Government, and provides the contractor with the greatest incentive for efficient and economical performance. [Ref. 64, 16.103(a)]

In selecting a contract type, a contracting officer should consider these factors: [Ref. 64, 16.104]:

- The degree of price competition present.
- The degree to which price analysis can provide a realistic pricing standard.
- In the absence of adequate price competition or if price analysis is not sufficient, the degree to which analysis of the contractor's estimated costs will provide a sound basis for negotiating prices.
- Type and complexity of the requirement.
- Urgency of the requirement.
- Period of contract performance.
- Contractor's technical capability and financial responsibility.
- Adequacy of the contractor's accounting system to support contract types other than firm-fixed-price.
- The impact of any concurrent contracts.
- The contractor risks due to the amount of subcontracting involved.

Considering these factors, the use of a firm-fixed-price (FFP) contract for the SOA PDSS presents a significant acquisition management issue for several reasons. First, it does not equitably distribute the cost risk involved in this complex effort. While this contract is for software "maintenance," much of the effort accomplished thus far has been enhancements, or modifications. These enhancements amount to new development, and therefore do involve considerable risk in estimating the amount of effort required to implement them. This risk was evident in the use of the 1675 SLOC limit in the PDSS base year. LORAL's insistence on the limit, and then vastly exceeding it indicates that even with its SOA experience, it is difficult for the contractor to accurately estimate effort for changes.

Another area of cost risk, discussed during the negotiations for the first option year of PDSS was in the flight testing of the annual software release. [Ref. 58] Cost risk in this area results from what LORAL believes are unclear test exit criteria, and Government requests to make corrections beyond the scope of the new changes that were implemented in the software. During the negotiations for the first option year, [Ref. 58] LORAL negotiators stated that in the past 96 percent of flight test STRs were in this category.

Second, as a sole source contract, the competitive forces of the market place are not in effect controlling prices. Contributing to this is that the Government lacks any significant price history on which to establish price reasonableness. As stated previously, because SOA is a unique system, cost comparison with other systems is difficult.

Third, software acquisition management requires increased buyer insight and visibility into the software development process. The use of a firm-fixed-price contract with its minimal administration does not support this increased visibility.

Fourth, a firm-fixed-price contract does not support the PM's goal to control the cost of PDSS. Under the firm-fixed-price contract, the Government has the ability to influence cost only once during contract performance, and that is at the negotiation table.

Finally, there appears to be confusion between the parties over exactly what the FFP portion of the contract buys. The Government sees the deliverable of the FFP portion of the contract being a software release. LORAL however, sees the FFP portion of the contract as a "framework" - what the researcher believes is a level of effort. This confusion may present problems later, especially in separating the costs of efforts accomplished under the FFP base effort, and any over and above efforts.

In the PDSS contract's base year, both the base contract line item for the IAP source release, and the over and above line items, were priced on a firm-fixed-price basis. In the first option year, however, the over and above line items were changed to a time and materials basis. The time and materials pricing arrangement provides for the acquisition of supplies or services at a fixed hourly direct labor rate which includes wages, overhead, general and administrative expenses, and profit. Materials are priced at cost plus material handling fee if appropriate. The incorporation of a ceiling price does limit the Government's risk. [Ref. 64, 16.601(a)] The integration of the new digital map processor will be accomplished on this over and above line. While this type of pricing arrangement does take into account the risk in estimating the effort required to accomplish the integration, it provides no incentive for the contractor to control cost. This is not the most appropriate pricing arrangement for this situation.

If these pricing arrangements are inappropriate, what are the options? The Federal Acquisition Regulation (FAR) addresses generally two broad categories of contract types, fixed-price contracts and cost-reimbursement contracts. These range from the presently used firm-fixed-price contract in which the contractor bears all of the cost risk to perform within the negotiated price, to the cost-plus-fixed-fee contract in which the Government bears the cost risk and the contractor's fee is fixed.

Fixed-price contracts are those in which the price is fixed, or in some cases may adjust based on certain circumstances. In a fixed price contract, the contractor is bound to deliver the specified goods or services for the negotiated price, regardless of his cost performance. The Federal Acquisition Regulation (FAR) Part 16.2 lists the following fixed price type contracts:

- Firm-Fixed-Price (FFP)
- Fixed-Price with Economic Price Adjustment (FPE)
- Fixed-Price Incentive, Firm Target (FPIF)
- Fixed-Price Incentive, Successive Targets (FPIS)
- Fixed-Price with Prospective Price Redetermination (FPRP)
- Fixed-Ceiling-Price with Retroactive Price Redetermination (FPRR)
- Firm-Fixed-Price, Level of Effort Term (FFP, LOE)

In his study of contracting for software within the Department of the Navy, Naval Postgraduate School student Henry Attanasio found that in software acquisition, typically four types of fixed-price contracts are used. [Ref. 71, p. 29]

- **Firm-Fixed-Price** - Under the FFP contract, the contractor is paid the negotiated price regardless of his cost experience in performing the work. It places on the contractor maximum risk and full responsibility for the resulting profit or loss. As such it also provides him with the maximum incentive to control costs and to perform in a manner which will yield the maximum profit.

This contract type requires minimal administration and is suitable for use in situations where the Government is acquiring *commercial products or those which can be easily priced by competition or by comparison with similar products*. [Ref. 64, 16.202]

- **Fixed-Price with Economic Price Adjustment** - The FPE contract allows for an upward or downward revision of the contract price based on either: changes in established or published prices of certain items; adjustments in the actual cost of labor or materials; and adjustments based on certain labor or material indexes specified in the contract. This contract type may be useful when there is serious doubt concerning the stability of labor or material prices during the period of performance. The adjustment contingencies should be limited to those which are beyond the contractor's control. [Ref. 64,16.203]

- **Fixed-Price Incentive** - The fixed-price incentive contract provides for adjusting profit and establishing the final contract price by application of a formula, or share ratio, which relates the final negotiated cost to a target cost. The final price is subject to a price ceiling which is negotiated at the start of the effort, and which serves to limit the Government's risk. There are two forms of the fixed-price incentive contract: the fixed-price incentive firm target (FPIF), and the fixed-price incentive successive targets (FPIS). In both forms, the parties negotiate a target cost, a target profit, a price ceiling, and a profit adjustment formula at the outset. In the FPIS contract the share ratio is applied at different times during contract performance, while in the FPIF form, the share ratio is applied at the end of the effort. Since the contractor's profit is tied to the share ratio, he has the incentive to control cost and thereby raise his profit. The FPIF contract is appropriate when the parties can negotiate at the outset a firm target cost, target profit, and a profit adjustment formula which will provide for an adequate incentive to the contractor. The FPIS contract is appropriate when available cost or pricing data is insufficient to support negotiation of firm target cost and profit before award, but that data will become available early in production. [Ref. 64, 16.403]

- **Firm-Fixed-Price, Level of Effort Term** - This contract type requires the contractor to expend a specified level of effort, probably stated in terms of labor hours, over a specified period of time, on work which is loosely defined, such as research. In return, the Government shall pay a firm-fixed-price for the contractor's effort. This contract type is suitable for research and usually results in the development of a report. However, payment is based on the effort expended rather than results of the effort. [Ref. 64, 16.207]

Cost-reimbursement contracts are for use only when the uncertainties in contract performance preclude estimating costs accurately enough to use one of the fixed-price contract types. Under cost-reimbursement contracts, the contractor is paid all allowable costs under the terms of the contract, up to a specified ceiling which he may not exceed (except at his own risk) without the approval of the contracting officer. Since the Government agrees to pay all allowable costs, it bears most of the cost risk. The following are the common cost-reimbursement contracts.

- **Cost-Plus-Incentive-Fee (CPIF)** - This cost-reimbursement contract provides for an initially negotiated fee which is then adjusted later based on the relationship between total allowable costs and target cost. The parties negotiate a target cost, a target fee, minimum and maximum fees, and a fee adjustment formula, or share ratio. After contract performance, the total fee is determined by application of the share ratio. The CPIF contract is appropriate for use in development or test programs in which a cost-reimbursement contract is appropriate and the parties can negotiate a target cost and a fee adjustment formula which will effectively motivate the contractor. [Ref. 64, 16.404-1]
- **Cost-Plus-Fixed-Fee (CPFF)** - This cost-reimbursement contract provides for the payment of a negotiated fee which is fixed at the inception of the effort. The fee is negotiated based on a percentage of the estimated costs and does not vary in relation to actual costs, however, it may be adjusted if the scope of work to be performed is expanded. As the contractor receives a fixed fee regardless of cost performance, this contract type provides minimal incentive for the contractor to control costs. It is appropriate for use in research or exploratory studies where the level of effort required is unknown. [Ref. 64, 16.306]
- **Cost-Plus-Award-Fee (CPAF)** - A CPAF contract is a cost-reimbursement contract which provides for the payment of a two part fee. The first part, called the base fee is fixed at the beginning of the effort and does not vary. The second part, called the award fee is an amount which the contractor may earn in whole or in part based upon the Government's judgmental, "after the fact" evaluation of the contractor's performance in terms of the criteria stated in the contract. These criteria may include areas such as cost control, quality, timeliness, management. The fee determination is made unilaterally by the Government and cannot be disputed by the contractor. [Refs. 64, 16.404-2 and 72, 216.404-2]

According to NASA, the agency which pioneered use of the CPAF contract, it is "...appropriate to use when cost uncertainty exists and key elements of performance cannot be objectively measured." [Ref. 73, p. 3] The contract is structured with evaluation criteria designed to motivate above minimum standards of performance established at the outset, and evaluation of performance by the Government at certain intervals. This evaluation process often includes feedback from the contractor. Additionally, the Government has the ability to shift emphasis among the evaluation criteria during performance of the effort. A 1992 Air Force study [Ref. 74] concerning the use of award fees in software acquisition concluded that the award fee is, "...the most flexible provision in the FAR and its supplements to influence contractor performance during the (software) development process." [Ref. 74, p. 80] Another important benefit of this contract type is the increased communication between the Government and contractor as a consequence of the award fee evaluation process. These benefits were confirmed by the researcher in interviews with personnel responsible for the administration of three CPAF contracts used for software development or maintenance. [Refs. 75 and 76]

A variation of the CPAF contract, is use of the award fee provision in conjunction with a fixed-price contract. This allows the Government to limit its risk with a negotiated fixed ceiling price, and also to motivate the contractor with the subjective evaluation of his performance. This variation was successfully used in a case involving the correction of software defects in the B-1B bomber. [Ref. 77, p. 26]

C. PROJECT MANAGEMENT ISSUES

Project management includes those activities which relate to the Government's management of the software development/maintenance process. The significant issues in this area are the lack of adequate project management resources, and use of software metrics.

1. Project Management Resources

Project management resources include systems and software engineering personnel, and the tools necessary to estimate software engineering effort. In the management of the PDSS program, the Program Manager essentially operates as a one man operation with whatever support he can muster through his personal initiative. This is an issue because he lacks the necessary technical support to adequately monitor the software development effort. This is especially critical in the area of analyzing the potential impact changes will have on the IAS. Without knowledgeable, organic systems engineering support, the Government is totally dependent on LORAL for analysis of the impact system changes will have on the IAS.

The second dimension of this issue is that currently, the PM does not have organic software engineering support. The lack of dedicated software engineering personnel in TAPO impacts project management in two respects. First, it forces the PM to personally manage the software development program. While the PM is an experienced SOA test pilot, he is not trained in software engineering. Additionally, the PDSS is but one of several programs which compete for his time. Second, without dedicated software engineering personnel, equipped with software cost estimation tools, the PM is unable to generate his own estimates for software development effort. This puts the Government at a great disadvantage when negotiating the firm-fixed-prices for the PDSS effort, or any over and above efforts. The TAPO organization has a billet for a software engineer, but a decision has been made not to fill this position.

2. Software Metrics

> When you can measure what you are speaking about and express it in numbers, you know something about it; but when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind: it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science.

This statement made by British scientist Lord Kelvin is appropriately displayed on the cover of the Navy's avionics metrics handbook [Ref. 78] The statement is certainly as timely now when discussing software project management, as when it was made over 100 years ago. In contracting for PDSS with LORAL, the PM is buying a "process" as much, if not more than he is buying a "product." Measurement of that process, and its product is one of the first steps toward the PM's goal of becoming a "smart buyer." Metrics provide the "hard" evidence [Ref. 19, p. 71] or indicators of performance. They serve to highlight problem areas that require management attention.

The lack of a metrics program impacts both the Government and LORAL. It is a significant issue for Government for several reasons. First, it sends the signal to LORAL that the Government's only concern in the PDSS contract is the end product, the delivered software. Second, it denies the PM from having any objective basis with which to evaluate changes or improvements in the software development process, and in the quality of the delivered software. Third, without metrics data, any complaints that the user or the PM may have concerning LORAL's software processes are largely just unsubstantiated opinions.

Lack of a metrics program is also a significant issue for LORAL. Without metrics, LORAL cannot objectively substantiate improvements it has made in the software process. For example, in the development of IAP release 12.0 (PDSS base year), LORAL began to incorporate aspects of the prototyping development methodology into the software process. While LORAL may view this as an "improvement" over a pure waterfall methodology, there is no way to measure the benefit without metrics data. This leads the researcher to ask the following questions: In what ways has this change improved the process? Have defects been reduced? Have specification changes been reduced? Has this change helped to lower the cost to the Government? Should the Government seek to expand the use of prototyping and if so why?

D. SOFTWARE ENGINEERING ISSUES

Software engineering is the technical and managerial process by which the software developer produces the products to be delivered under the contract. It consists of specific methods, procedures, and tools which assist in that process. There are three software engineering issues impacting the SOA PDSS program. They are the software development toolset and language, requirements definition and the software development methodology, and software process maturity and process improvement.

1. Software Development Toolset and Language

The SOA IAP software was developed in JOVIAL, which is one of the older Higher Order Languages (HOL) used for computer programming. The toolset on which it was developed and is now maintained is also old technology. As a result of the age and limited capabilities of the toolset, functions which could be automated, are still done manually. Software configuration management is an example. Modern Computer Aided Software Engineering (CASE) environments for the DoD standard programming language, Ada allow for automating functions such as configuration management, documentation generation, and testing. Automation of these functions can significantly reduce the engineering labor hours involved in the software development process.

2. Requirements Definition and the Software Development Methodology

The waterfall development methodology in use at LORAL represents a significant issue when linked with a firm-fixed-price contract. Because it is difficult for the contractor to adequately define the detailed requirements for the given software product up front, there is significant cost risk under a firm-fixed-price contract. The fact that establishing the detailed requirements is difficult is recognized both by the software industry and by DoD.

In his address to the annual DoD Software Technology Conference, Lieutenant General Otto J. Guenther, the Army's software executive made these remarks concerning software requirements:

> Where do I see opportunities for software improvement? ...allow requirements to evolve through the acquisition process. I don't think we can get the requirements 100 percent right initially in the acquisition process. What we need are adaptable requirements and an acquisition process for software intensive systems. [Ref. 79, p. 5]

One way to incorporate adaptable requirements into the software acquisition process is through prototyping. Prototyping is essentially the process of building a working replica of the software system. It may be used in conjunction with the waterfall methodology to demonstrate technical feasibility. In this sense it is also helpful in understanding and extracting the ***exact*** user requirements. The goal is to limit cost by understanding the problem before committing additional resources. [Ref. 80, p. 15]

LORAL made use of the prototyping methodology to evolve requirements in the base year of PDSS. In the first option year, they also plan to use it to define the detailed requirements for the integration of the digital map processor. This effort is priced on a time and materials basis. Therefore any savings achieved from using the prototype method will be passed along to the Government in the form of reduced overall labor hours. However, a significant portion of the contract is firm-fixed-price, and any savings achieved through incorporation of prototyping on this contract line item will not pass to the Government.

3. Software Process Maturity and Process Improvement

The lack of a formal assessment of LORAL's software process maturity, and a resulting process improvement plan are significant issues for the Government. The maturity of LORAL's software processes in use on the SOA program have never been independently assessed by the Government. As result, the Government lacks a quantifiable standard, such as the SEI CMM by which to measure the maturity of LORAL's process. In conjunction with this, there does not appear to be a formal process improvement plan specifically addressing the SOA software processes. These are issues because the use of process assessments, and the process improvement plans which are derived from them, are called for in policy [Ref. 13 p. 6-D-1-1] and have been proven to be effective in reducing the cost of software development and maintenance.

For example, a recent article in *Cross Talk, The Journal of Defense Software Engineering* [Ref. 81, pp. 14-17] illustrated the economic benefits of software process improvement. The article discusses a study which was performed by Software Productivity Research Incorporated in the test software branches of the Air Force's

Oklahoma Air Logistics Center, Software Division. The purpose of the study, commissioned by the Air Force's software executive, was to quantify the economic benefits of investing in software process improvement. Four projects, which spanned from 1986 to 1995, were included in the study. The study concluded that a process improvement investment of $1.5 million over an eight year period, yielded a cost savings of $11.3 million, a return on investment of 7.5 to 1. The benefits were achieved in these areas:

- 90 percent reduction in defect rates.
- 26 percent reduction in the average cost of a maintenance action over the last two years.
- Productivity increased to levels ten times that of the baseline project.

This article clearly demonstrates the economic benefits of process assessments and of investing in process improvement.

E. LESSONS LEARNED

As a result of the analysis of the SOA PDSS case, the following lessons have been learned:

- The cost of software development, whether new development or PDSS is a function of factors influenced by both the Government and the contractor. As such, costs must be controlled through actions on the part of both the Government and the contractor.
- Documentation is expensive. The type and amount of documentation purchased by the Government should reflect what is needed to support the future strategy and needs of the program.
- Even in the absence of a formal acquisition strategy, the pattern of actions and decisions made in a software-intensive program can indicate the selection of a certain strategy. Decisions made throughout a development program have "downstream" impacts.
- The CRLCMP is the key management plan during the operations and maintenance phase of the software lifecycle. It should be maintained and updated to reflect the current and future PDSS concept.

- System specific knowledge and expertise are critical in maintaining software-intensive systems.

- Transition from one source to another for software maintenance is risky. It requires preparation and planning.

- The PPBS process requires that PDSS requirements be planned at least two years in advance. Failure to properly forecast requirements leaves a program with inadequate funds to implement desired changes.

- The development and maintenance of software-intensive systems creates a situation in which the Government and contractor are dependent on one another. The degree of cooperation in the relationship should reflect this mutual dependence.

- The use of a firm-fixed-price contract for complex weapon systems software development and maintenance is inappropriate. Weapon systems software development or maintenance requires the use of contracts which adequately share cost risk, provide cost visibility, and provide incentives for the contractor to control cost.

- The use of the SLOC metric is inaccurate for effort estimation.

- The award fee incentive, whether used in a cost-reimbursement or fixed-price contract, provides a flexible and effective tool for motivating contractor performance. It also serves to improve communication between the Government and the contractor.

- Proper management of software acquisition requires resources. These resources include systems and software engineering personnel, and software cost estimation tools.

- Metrics provide the PM with valuable insight into the products and processes involved with software development. Lack of a software metrics program limits the PM's insight into the contractor's software process.

- The age and capabilities of a program's software development tools contribute to some degree to the cost of the development or maintenance effort. Modern CASE tools can reduce software engineering labor costs through automation of functions such as configuration management and testing.

- To be a true cost reduction benefit to the Government, use of the prototyping development methodology must be linked with a contract type which shares savings with the Government.

- Software process assessments such as the SEI CMM provide the Government with insight into the maturity of a contractor's software process. These assessments in turn provide the basis for process improvement plans. Investments made in process improvements yield future cost savings.

F. CHAPTER SUMMARY

This chapter has provided the identification and the analysis of the significant issues impacting the management of the SOA PDSS software acquisition. This analysis has revealed that the significant management issues with regard to the SOA PDSS are in the areas of acquisition management, project management, and software engineering. These issues are significant in that failure to address them hinders the PM in his efforts to become a "smart buyer", and to properly manage the PDSS program.

V. CONCLUSIONS AND RECOMMENDATIONS

A. INTRODUCTION

The primary purpose of this chapter is to provide the conclusions and recommendations for this investigation. It consists of four major sections. The first section presents the conclusions drawn from the analysis of the issues identified in the SOA PDSS case, and the lessons learned. The second section consists of specific recommendations to address the issues. The third section discusses the answers to the research questions. Finally, areas for further research are provided.

B. CONCLUSIONS

The following are the conclusions drawn from the analysis and lessons learned.

1. General

This thesis confirms the findings of the GAO [Ref. 2], the Defense Science Board [Ref. 10], and author Robert L. Glass [Ref. 9]. These and numerous other reports and articles concerning DoD software have emphasized that the key to success in weapon system software acquisition is effective ***management*** by PMs. Effective software management requires the use of highly skilled personnel resources, tools, and methods. Perhaps the most essential element of effective program management is the support of senior leaders who are well informed in the complexities of software development. These leaders must understand that software management is vastly more difficult than hardware management. The reasons for this difficulty, presented in Chapter II include:

- The lack of adequate software management concepts, methods, and practices.
- The complexity of mission critical software systems.
- The invisibility of software. It is difficult to measure its essential characteristics and those of the software process.
- The difficulty in defining software requirements.
- Software is changeable, flexible and modifiable. Because it has these qualities, control of the modification process is integral to the effectiveness of software.

In the absence of effective software management, these difficulties often dominate software acquisition. Especially important is management of software systems as they evolve during PDSS. Without the proper resources mentioned previously to support effective management of PDSS, the software which now controls modern weapon systems may not be ready when called upon to perform its mission.

2. Acquisition Management

Analysis of the acquisition management issues has revealed the following: First, that the Government sees cost as the underlying issue in the SOA PDSS case, with their perception being that LORAL is too costly. However, the researcher concludes that the true issue in this regard is not the magnitude of the cost, but rather how the cost is managed by both the Government and LORAL. Clearly, the Government needs to do more to manage the cost it believes is excessive than simply awarding a firm-fixed-price contract which does nothing more than place an upper limit on cost. If the PDSS cost is genuinely a Government concern, its elements should be analyzed in detail and strategies to reduce it must be developed jointly with LORAL. In addition, because cost is such an important issue from the Government's perspective, it is in LORAL's interest to take sincere actions to evaluate ways to reduce it. Because the major costs are directly associated with manpower, this will inevitably require LORAL to reduce staffing on the project. Failure to find such economies, may result in the loss of the SOA business.

Second, program plans are essential to the effective management of the program. Currently the only planning mechanism in use on the SOA PDSS program is the contract SOW. Without the framework which a program plan provides, management is *ad hoc,* an exercise in *crisis management,* not program management. The most logical planning mechanism to use for PDSS is the SOA CRMP. Lack of a current CRMP leaves the PM without the framework necessary to plan the future of the PDSS program, and it fosters a sort-term perspective. This short-term perspective is evident in two primary areas. First, is the mistaken belief in the user community that a competitive procurement of PDSS following this contract is a genuine possibility. With regard to this issue, the researcher concludes that because system knowledge and experience are so critical in maintaining complex, integrated software systems, LORAL is *realistically* the only source capable of performing PDSS. Second, the program lacks a computer resources master schedule which links planned IAS enhancements to the budget process. While SIMO is budgeting for enhancements as it prepares budget estimates, there is not a single integrated plan which documents this. Lack of such a plan leaves the PM without visibility into the future requirements and a budget to implement those requirements. In the first two years of the PDSS program, lack of such a plan left the PM with budgets which appeared to be tight in relation to the cost to implement the desired enhancements.

Third, the Government and LORAL are dependent on one another in this relationship. The level of interdependence in the buyer-seller relationship clearly dictates a closer, more cooperative relationship. Fourth, the use of a firm-fixed-price contract does not support the PM's goal of becoming a smart buyer. It limits the Government's visibility of the actual cost of PDSS, and provides no direct incentive to LORAL to reduce the overall contract cost. The award fee incentive is perhaps one way to assist the PM in achieving his goal. It is a flexible motivation tool and it enhances communication between the parties.

3. Project Management

The analysis of the SOA PDSS case revealed that the two significant issues in regard to project management are project management resources and software metrics. Analysis of these issues revealed the following. First, management of weapon systems software acquisition requires specific resources. The PM currently lacks the resources needed to properly manage the complexities of software acquisition. These resources include systems and software engineering personnel with system knowledge, and automated tools to estimate software costs. Even if there was to be a change in PDSS contractors in an effort to find a lower price, the PM is still not properly equipped to manage them. Second, management of software acquisition also requires metrics. Without the use of software metrics in the SOA PDSS program, the PM is blind to the essential details of LORAL's software process, quality, and productivity.

4. Software Engineering

The area of software engineering presents three significant management issues. These issues are:

- The age of the software development tools and language,
- Requirements definition and development methodology, and
- Software process maturity and process improvement.

Analysis of these issues revealed first that the age of the software tools and the JOVIAL language, for which CASE technology is not available, is contributing to some degree to the perceived high cost of PDSS. Second, while LORAL wants to expand the use of prototyping to evolve exact software requirements, any savings achieved through prototyping would not be shared with the Government under the current firm-fixed-price contract. Finally, the Government has never formally assessed the maturity of LORAL's software processes. Without a formal assessment of LORAL's processes, the maturity of those processes in relation to a known standard such as the SEI's CMM, is unknown. Also, without a formal process assessment, there is no basis for a detailed process improvement plan.

C. RECOMMENDATIONS

Based on the analysis in Chapter IV of this thesis, and the conclusions of the previous paragraphs, the following recommendations are provided. These recommendations seek to improve the management of the SOA PDSS program.

1. Provide the PM with the necessary software management resources.

If TAPO is to continue management of the PDSS program, it must be equipped with the resources to do so. These resources should include a systems engineer, a software engineer, and the automated project management and cost estimation tools required to perform basic management functions. The process of gathering these resources should begin by filling TAPO's now vacant software engineer position. These resources will support the PM's capability to perform project management functions which mirror those of LORAL. These functions include assessment of the system impact of proposed enhancements or corrections, cost estimation of enhancements and corrections, analysis of metrics data, and configuration management. Without these minimum management resources, it is doubtful that the cost of PDSS will ever be properly managed, regardless of the contractor. An annual contract of approximately $4 million requires Government management oversight.

2. Develop a more cooperative relationship with LORAL

The pattern of actions taken in regard to PDSS indicates a strategy which includes LORAL as the SOA systems integrator. As stated previously, LORAL is currently the only source that can realistically perform PDSS on the complex SOA system. This strategy should be formalized in the SOA CRMP and form the basis of a closer, more cooperative relationship with LORAL for a second three year contract. One of the keys to a more cooperative relationship will be a contract type which is fair and meets the objectives of both parties. One of LORAL's objectives in particular is to maintain stability in the SOA workforce - to have a predictable backlog. This was in essence the purpose of the "framework" established by the base effort of the first option year. One of the Government's objectives for the contract is the ability to put new requirements (such as digital map integration) into the contract effort at any time. In addition to these particular objectives, the new contract should provide the following: a more equitable sharing of cost risk, Government visibility as to the true costs of PDSS, a positive incentive for LORAL to reduce costs, and, a vehicle for more open communication.

One particular contract form which may assist the parties in meeting mutual goals is the task order contract. While not one of the particular contract types discussed in

Chapter IV, it may include provisions for awarding task orders priced in several different ways. A task order contract is defined as:

> ...a contract for services that does not procure or specify a firm quantity of services (other than a minimum or maximum quantity) and that provides for the issuance of orders for the performance of tasks during the period of the contract. [Ref. 82, p. 12]

The task order contract should be structured with a minimum and maximum on engineering labor hours. The minimum should be awarded annually on a CPAF basis to maintain a minimum level of staffing working on SOA software issues for the entire year of support. This would provide LORAL the consistency it needs in planning project staffing. It also provides the Government with additional insight and control it needs. Additionally, the increased communication required by the award fee evaluation process will be key in development of a more cooperative relationship. The suggested award fee criteria would address the following areas:

- Cost control.
- Program Management.
- Technical.
  - Software process improvement.
  - Software testing and quality.
  - Software metrics program.

As the Government generates additional requirements during the period of performance, additional tasks may be awarded. These can be priced by the most appropriate means for the task, in accordance with the contract types specified in the terms of the contract. Simple tasks which are low risk and easily priced may be firm-fixed-price. Complex integration efforts involving higher risk may be priced on a CPAF basis.

The use of a task order contract will facilitate a cooperative relationship because the award of an annual minimum effort will show a commitment to LORAL on the

Government's part. It will also provide the Government the flexibility to meet new requirements as they become known. One particular task the Government should consider awarding is a study to further define what would be required to transition to another source (either a Government SSA or another contractor) in the future. The Government may ultimately decide to stay with LORAL for the life of the program. However, the PM should continue to evaluate future PDSS options. A generic listing of transition considerations is provided in Appendix B of this thesis.

3. Reestablish a system of program planning through the SOA CRMP

The program's planning needs can be fulfilled in a single plan, the SOA CRMP. The PM should initiate actions to update CRMP to reflect the new PDSS concept. Formalizing the strategy in the CRMP will serve to solidify a commitment to LORAL and provide the much needed long-range direction for the program. Special attention should be paid to the development of a computer resources master schedule which forecasts requirements a minimum of two years into the future. The concept behind this schedule, depicted graphically in Figure 20, is to establish a link between planned requirements, cost estimates of the implementation of those requirements, and the SOA PDSS budget. Planning to this level of detail requires the PM to have the resources discussed earlier. Without a planning process such as this, reflected in a schedule, PDSS will always appear to be expensive, because the budget will not support system requirements. A schedule of this nature will also serve to provide visibility to both the PM and the user of the cost of PDSS. It will assist them in prioritizing requirements and in making the tradeoffs required when budgets are relatively fixed and thus a constraining factor.

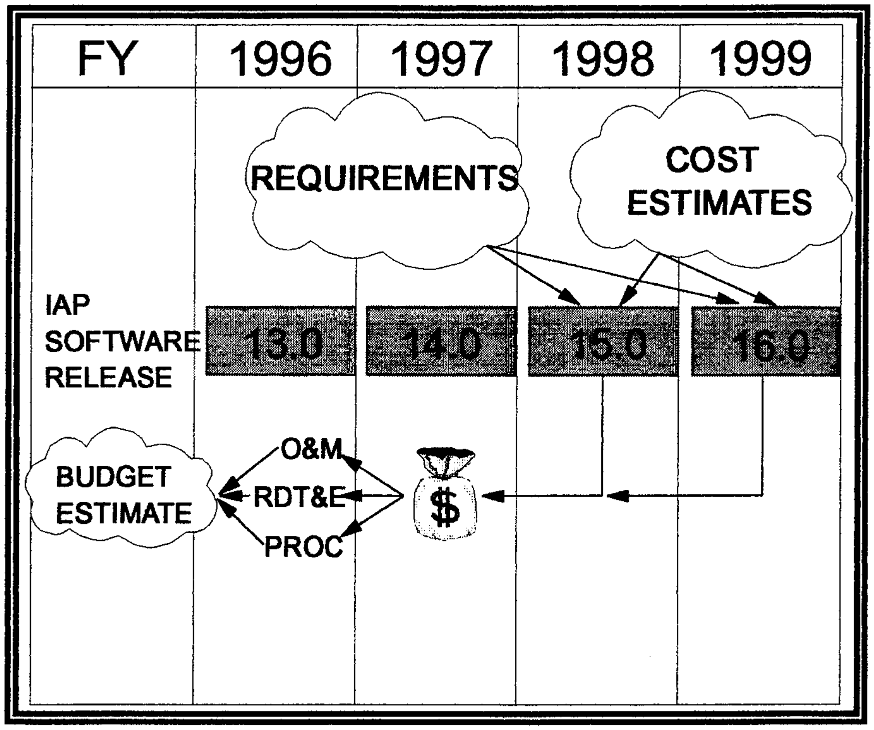

Figure 20. Computer Resources Master Schedule

4. Initiate a software metrics program

One of the keys to meeting the PM's goal of becoming a "smart buyer" is to increase insight into LORAL's software development process. Perhaps the most effective way to increase insight is to gather and analyze pertinent metrics data. To be a truly useful benefit, which will warrant the additional effort and expense involved in gathering and analyzing the data, the selected metrics must address issues which concern both the PM and the user. A particularly useful method to define the metrics set is the Goal/Question/Metric paradigm created by Victor Basili. [Ref. 83] This method has been effectively applied in the metrics program at Hewlett-Packard. [Refs. 21, 84]

The principle behind the paradigm is that each software project has a set of goals. For each goal, there is a set of questions that may be asked to determine if those goals are being achieved. Many of these questions can be answered with measurable metrics data. The relationship between the goals, questions, and metrics are shown in Figure 21.

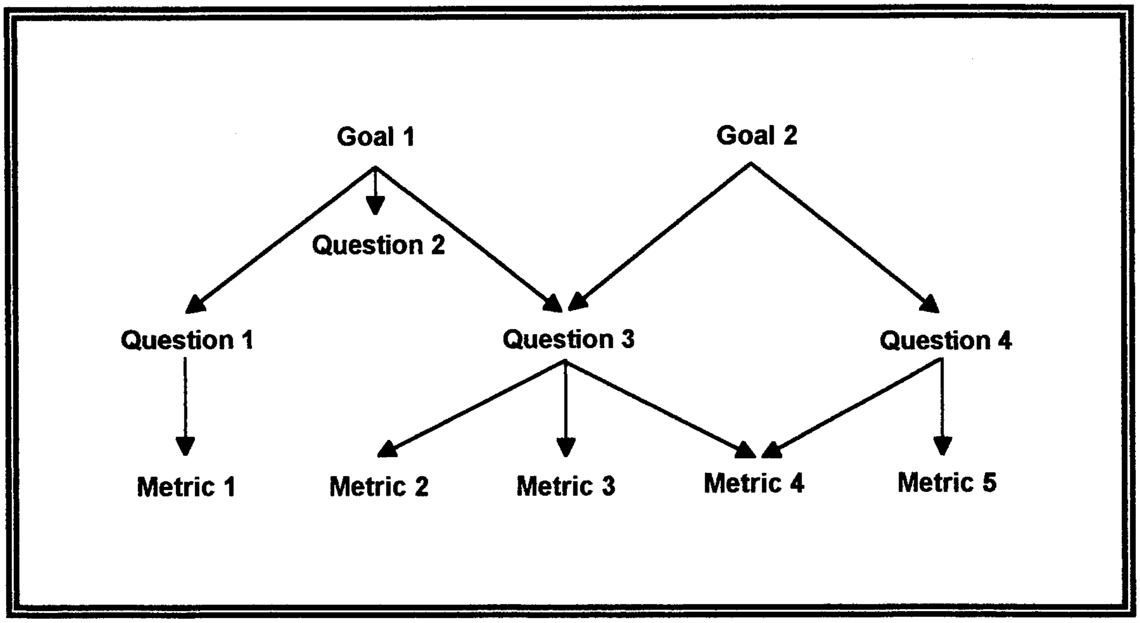

Figure 21. The Goal/Question/Metric Paradigm
From [Ref. 21]

Using this method, one of the goals of the Hewlett-Packard metrics program is to *minimize engineering effort.* In the SOA PDSS case, since the Government feels that PDSS costs too much with LORAL, minimizing engineering effort may also be one of their goals. Several of the questions and corresponding metrics which Hewlett-Packard uses to achieve this goal are:

- Question: Where are the resources going? Where are the worst rework loops in the process?
  - Metric: Engineering months by product/product component/process activity.
- Question: How much do the maintenance phase activities cost?
  - Metric: Engineering time and cost.
- Question: What are the major cost components? What aspects affect the cost?
  - Metric: Engineering months by product/product component/process activity.
- Question: How do costs change over time?
  - Metric: Track cost components over the entire maintenance lifecycle.

- Question: How maintainable is the software product as changes occur? When do I give up and rewrite?
  - Metrics: Incoming problem rate.
  - Defect density.
  - Code stability.
  - Code complexity.
  - Number of modules changed to fix one defect.
- Question: What will the maintenance requirements be?
  - Metrics: Code stability, complexity, size.
  - Prerelease defect density.

The exact goals, questions, and metrics are issues which need to be defined jointly by the user, the PM, and LORAL. The point is that to be a "smart buyer" and to begin making progress toward program goals, the program needs a metrics program. The researcher recommends beginning a metrics program gradually with a single metric. This metric should track estimated versus actual engineering effort by product and by phase of the engineering process. This metric will serve two purposes. First it will provide insight as to where development resources are going. Second it will assist the Government in developing insight as to the accuracy of LORAL's cost estimation system. In further defining a metrics program, the PM should make maximum use of LORAL's current metrics system. Other sources to consult in developing a metrics program include the Army's Software Test and Evaluation Panel (STEP) metrics program, and the Air Force's Guidelines for Successful Acquisition and Management of Software Intensive Systems [Ref. 20]

5. Conduct a formal software capability evaluation of LORAL's SOA process

The researcher recommends that the PM conduct a formal assessment of the maturity of LORAL's SOA processes. A formal assessment will serve two purposes. First it will provide the Government with valuable insight into the details of LORAL's software development process. This benefit would be maximized if the assessment could be conducted by trained Government software engineering personnel. If not, the PM should consider the use of a consulting firm such as Software Productivity Research to conduct such an assessment. Second, the assessment will serve as the basis for the development of a process improvement plan which will seek to maximize the performance of the SOA software process. This may also require the Government to share in the cost of implementing such a plan.

D. ANSWERS TO RESEARCH QUESTIONS

This section provides summarized answers to the questions which guided this research.

1. Primary Research Question

The primary research question for this thesis is: **What are the significant management issues associated with the current SOA PDSS program and what actions can be taken to address these issues?**

Analysis of the SOA PDSS case revealed that the significant management issues are found in the areas of acquisition management, project management, and software engineering. These issues can be addressed through actions which require the Government to better manage software acquisition. The details of such actions were previously discussed.

2. Subsidiary Research Questions

a. What is PDSS and what are its essential elements?

PDSS is the process of ensuring that fielded weapon system software continues to meet the needs of the user. It includes two categories of software "maintenance" activities. These categories are corrective maintenance, activities focused on the correction of software defects; and enhancements, activities performed to either accommodate changes in the operational environment or to improve the software's performance. PDSS requires detailed planning prior to deployment of the software system, and a structured software engineering approach similar to that required during software development.

b. What are the principal policies and guidance concerning PDSS?

The principal policy concerning PDSS of Army weapon system software is provided by the draft policy on PDSS. This policy seeks to improve the planning and preparation for PDSS, and to centralize at the DA level the funding and priorities for PDSS. The only DoD guidance concerning how to conduct PDSS is given by Military Handbook 347, Mission-Critical Computer Resources. This handbook addresses both the preparation for PDSS, and the conduct of PDSS.

c. What are the elements of the current SOA PDSS program?

The SOA PDSS program consists of the key players, interacting in the three processes involved in software acquisition. The key players are the user, the 160th Special Operations Aviation Regiment (Airborne); the PM, a project officer at the Technology Applications Program Office; and the contractor, LORAL Federal Systems Company of Owego, New York. In the acquisition of PDSS for the SOA aircraft, these players interact in three processes. These processes are acquisition management; project management; and software engineering.

d. What are the significant management issues associated with the current PDSS program?

Analysis of the SOA PDSS case revealed that the significant management issues are the following:

- **Acquisition Management Issues**
  - Cost
  - Program Plans

- The Buyer-Seller Relationship
- Contract Type

- **Project Management Issues**
  - Project Management Resources
  - Software Metrics
- **Software Engineering Issues**
  - Software Development Toolset and Language
  - Requirements Definition and Development Methodology
  - Software Process Maturity and Process Improvement

e. What actions can be taken to address these issues?

In response to these issues, the researcher recommends the following actions:

- The PM be provided with the necessary software management resources. These resources include systems and software engineering personnel, and automated tools to assist in developing software cost estimates.
- The PM should formalize a PDSS concept and strategy which includes a cooperative relationship with LORAL.
- The PM should reestablish a system of program planning through the SOA CRMP.
- The PM should initiate a software metrics program.
- The PM should conduct a formal software capability evaluation of LORAL's SOA process.
- This evaluation should lead to the development of a process improvement plan.

f. What lessons can be learned from the SOA program's experience with PDSS?

The principal lessons learned from the SOA PDSS case are:

- The cost of PDSS is a function of factors on both sides of the buyer-seller relationship. As such it must be addressed through actions on both sides of the relationship.

- PDSS requirements must be planned in advance so that the budget will support the purchase of the desired functionality.
- The firm-fixed-price contract is not appropriate for PDSS of complex weapon system software. The selected contract type must adequately share risk, provide visibility as to the true cost of software maintenance, and provide incentives to reduce costs.
- Effective management of software acquisition requires adequate resources. These resources include systems and software engineering personnel and software cost estimation tools.

E. AREAS FOR FURTHER RESEARCH

The following areas are recommended for further research:

1. A Software Management Capability Model

Development of tool and model to assess the software acquisition management capabilities of DoD program management organizations. This thesis confirmed the findings of the GAO [Ref. 2] and the Defense Science Board [Ref. 10]. These reports concluded that despite the recent improvements in software engineering methods and technology, effective management is a major key to success in software acquisition. A tool of this nature would be used to assess the particular strengths and weaknesses of a program management organization's software management capability. It would address personnel qualifications, tools, and methods. The model would also serve as the basis for management process improvement plans.

2. Implementation of the Army Policy on PDSS

The Army's draft policy on PDSS seeks to centralize the priorities and funding for PDSS of software-intensive systems. This study would evaluate the impact of the policy on weapon systems readiness, and the ability of users to implement desired enhancements.

3. Pre-Deployment Software Support of the RAH-66 COMANCHE

A study of the pre-deployment software support preparations for the Army's RAH-66 COMANCHE helicopter program. The SOA program represents one of the

Army's first experiences with highly integrated avionics systems. The SOA's 380,000 source lines of code are dwarfed in comparison to the COMANCHE's estimated one million lines of code. This study would examine the COMANCHE's pre-deployment software support activities. Specifically it would investigate the PDSS concept, and whether the PM is prepared to implement that concept.

4. The Army's Systems and Software Engineering Capabilities

A study of the systems and software engineering capabilities of the Army. Army weapon systems continue to increase in complexity and software content. As they do, technical personnel with system specific expertise will be critical to the Army's ability to manage these programs effectively. This thesis has shown that this may be an area which impacts other Army programs. A study of this nature would survey the Army's current systems and software engineering capabilities. It would determine if the Army has sufficient technical personnel, and if not, how the situation could be addressed.

5. SOA PDSS

This thesis has examined the SOA PDSS case for only the base, and first option years of the first PDSS contract. A follow-on study would examine the SOA PDSS program to determine if lessons learned have been applied in later contracts.

F. CONCLUDING REMARKS

In software acquisition, there are no "silver bullets". No single method or action will address all of the issues involved in a software acquisition management situation as complex as the SOA PDSS. It is doubtful that merely changing the contract type, or beginning to gather metrics data, or even changing sources, alone would "solve" the SOA PDSS "problem." Rather, solving the problem requires a management commitment to trying a combination of these methods, and a commitment to change methods when required. The issue is whether we as DoD managers manage our software issues or let the issues manage us.

APPENDIX A. ACRONYM LIST

ACAT	Acquisition Category
ACs	Airframe Contractors
AHASS	Army SOA Helicopter Avionics System Simulator
ANVIS	Aviator's Night Vision Imaging System
ASE	Aircraft Survivability Equipment
ATCOM	U.S. Army Aviation and Troop Command
AVSCOM	U.S. Army Aviation Systems Command
BSAS	Boeing-Sikorsky Air Services
CASE	Computer Aided Software Engineering
CCB	Configuration Control Board
CDB	Common Database
CDU	Control Display Unit
CECOM	U.S. Army Communications-Electronics Command
CMFD	Color Multifunction Display
CMM	Capability Maturity Model
CMS	Combat Mission Simulator
CPAF	Cost-Plus-Award-Fee
CPFF	Cost-Plus-Fixed-Fee
CPIF	Cost-Plus-Incentive-Fee
CRLCMP	Computer Resources Life-Cycle Management Plan

CRMP	Computer Resources Management Plan
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSU	Computer Software Unit
DA	Department of the Army
DCE	Decompression Engine
DCSOPS	Deputy Chief of Staff Operations
DLSIE	Defense Logistics Studies Information Exchange
DMP	Display Management Program
DoD	Department of Defense
DP	Display Processor
DPML	Display Processor Memory Loader
DTIC	Defense Technological Information Center
DTS	Data Transfer System
DTT	Desk Top Trainer
ECP	Engineering Change Proposal
EUT&E	Early User Test and Evaluation
FAR	Federal Acquisition Regulation
FFP	Firm-Fixed-Price
FFP, LOE	Firm-Fixed-Price, Level of Effort Term
FLIR	Forward Looking Infrared

FPE	Fixed-Price with Economic Price Adjustment
FPIF	Fixed-Price Incentive, Firm Target
FPIS	Fixed-Price Incentive, Successive Target
FPRP	Fixed-Price with Prospective Price Redetermination
FPRR	Fixed-Ceiling-Price with Retroactive Price Redetermination
FRR	Flight Readiness Review
GAO	General Accounting Office
GPS	Global Positioning System
GSS SLD	Ground Support System Software Load Device
GTR	Government Trouble Report
HF	High Frequency
HOL	Higher Order Language
IAP	Integrated Avionics Program
IAS	Integrated Avionics Subsystem
IEEE	Institute of Electrical and Electronics Engineers
ILS	Integrated Logistics Support
INU	Inertial Navigation Unit
IOT&E	Initial Operational Test and Evaluation
IPR	In-Process Review
IV&V	Independent Verification and Validation
LCCS	Life Cycle Contractor Support

LCSE	Lifecycle Software Engineering
LOC	Lines of Code
LRIP	Low Rate Initial Production
LRU	Line Replaceable Unit
MCCR	Mission Critical Computer Resources
MDG	Map Display Generator
MDG ML	Map Display Generator Memory Loader
MEP	Mission Equipment Package
MFP	Major Force Program
MICOM	U.S. Army Missile Command
MMFD	Monochromatic Multifunction Display
MMP	Map Management Processor
MMR	Multimode Radar
MMS	Mission Management System
MP	Mission Processor
MY	Man Year
NASA	National Aeronautics and Space Administration
NDI	Non-developmental Item
ODISC4	Office of the Director Information Systems for Command, Control, Communications, and Computers
OFP	Operational Flight Program

O&M	Operations and Maintenance
PAT	Process Action Team
PBIT	Power-On-Built-In-Test
PDSS	Post Deployment Software Support
PEO	Program Executive Officer
PIDS	Prime Item Development Specification
PM	Program Manager
PM SOA	Product Manager, Special Operations Aircraft
PPBS	Planning, Programming, and Budgeting System
PTT	Part Task Trainer
RDT&E	Research, Development, Test and Evaluation
ROC	Required Operational Capability
RTU	Remote Terminal Unit
RTU ML	Remote Terminal Unit Memory Loader
SATCOM	Satellite Communication
SEE	Software Engineering Environment
SEI	Software Engineering Institute
SGP	Signal Generator Program
SIMO	Systems Integration Management Office
SINCGARS	Single Channel Ground and Airborne Radio Subsystem
SLOC	Source Lines of Code

SOA	Special Operations Aircraft
SOF	Special Operations Forces
SOF MOD	Special Operations Forces Helicopter Modification
SOW	Statement of Work
SROMP	Start-Up Read Only Memory Program
SRS	Software Requirements Specification
SSA	Software Support Activity
STR	Software Trouble Report
STRICOM	Simulation, Training, and Instrumentation Command
TAPO	Technology Applications Program Office
TF	Terrain Following
TIM	Technical Interchange Meetings
QA	Quality Assurance

APPENDIX B. SOFTWARE TRANSITION CONSIDERATIONS

A. PURPOSE

This appendix describes the factors which must be considered in the transition of software support from one activity to another. Much of the literature concerning software support transition assumes that transition occurs between a commercial developer and a Government software support activity (SSA). However, similar procedures would be required in the orderly conduct of transition from one commercial contractor to another.

B. DEFINITION

Military Handbook 347, Mission-Critical Computer Resources Software Support, defines software transition as: [Ref. 25, p. 10]

> A ***controlled*** and ***coordinated*** sequence of actions wherein software development passes from the organization performing initial software development to the organization performing PDSS. Software transition also involves the SSA moving from pre- to post-deployment software support activities.

The researcher has emphasized ***controlled*** and ***coordinated*** because the capability of the new support activity (whether Government or contractor) to properly support the software depends on what occurs or fails to occur during transition. Transition should not be conducted hastily. In fact software maintenance expert Thomas M. Pigoski believes that a good transition should take place over a two to three year period. [Ref. 31, p. 47] Implied in this definition is the fact that during transition, the PM will be managing the coordinated activities of two software organizations. Also implied are the increased costs the Government will bear while paying for the activities of two organizations during the transition.

C. TRANSITION CONSIDERATIONS

The following sections discuss the major considerations to be addressed during the transition of software support from one organization to another.

1. Transfer of System Knowledge

Because system specific knowledge is so critical in software development and maintenance, this is perhaps the most important consideration in software transition. The system specific expertise resident in the developer must be transferred to the support organization during transition. The considerations discussed in the sections below support the transfer of system knowledge in a controlled and coordinated manner.

2. Management and Administration Procedures

The goal of these procedures is to ensure that the necessary management structure is in place to control the transition. This includes a memorandum of agreement between the developer and the SSA which outlines specific responsibilities. If the transition is to occur between two contractors, an acquisition plan which defines the specific contracting approach will be critical to a coordinated transition from one contractor to another. Included in this consideration is the update and transfer of program plans such as the project management plan, configuration management plan, software development plan, and quality and test plans. Together, these plans will define for the new support activity how the system is supported and what tools are required to maintain the software.

3. Software Engineering and Test Procedures

This consideration addresses the software engineering environment and the development procedures used in maintaining the software. For a program such as SOA, which was developed on software tools which are 1980s technology, transition may be the logical point at which to introduce more modern tools. According to researcher Dr. Thomas Vollman, specific consideration should be given to tools which can accomplish the following: [Ref. 29, p. 194]

- Provide management insight into development progress.
- Provide quantitative metrics assessments of aspects of the software system as designed and built.
- Define test requirements for upgrades based on specification changes.

- Provide measures of how well the software has been tested.
- Provide for automation of configuration management activities. These include code management and control, and software builds and version descriptions.
- Provide support for engineering changes to the software, including describing the structure of the code, and documentation of those structures.

4. Training and Personnel

Perhaps the most critical consideration in the transfer of knowledge during transition is the training of the software maintenance personnel. As discussed previously, system specific knowledge is key in maintaining software systems. This knowledge transfer will take time. It must be accomplished gradually, from one activity to another through a program of "hands-on," on-site training. Personnel from the organization assuming the support role must be intimately involved in the software development and maintenance process of the development organization as soon as possible. As the new organization's software support environment is established, personnel from the development organization should then be involved in training personnel at the new site. Pigoski calls these individuals "maintenance escorts." [Ref. 31, p. 48] This training culminates with the development, testing, delivery, and acceptance by the user of a new software release.

5. Technical Documentation Procedures

Large programs often have voluminous documentation. Often this documentation is out of date. The transition process should include a plan for the systematic update of system documentation and the transfer of the Technical Data Library from the developer to the maintainer. In addition to the plans already addressed, the key documents to consider are the following:

- software requirements specifications
- software design documents
- software product specifications
- version description documents

D. TRANSITION REFERENCES

The following references are provided to assist the PM in further examination of the issues involved in transition of software support:

- Department of Defense, MIL-HDBK-347, *Military Handbook: Mission-Critical Computer Resources Software Support*, 22 May 1990.
- Pigoski, Thomas, M., *Life Cycle Strategy: Software Support on the Front Line*, Software Maintenance News, Inc., 1995.
- Vollman, Thomas, "Transitioning From Development to Maintenance," *Proceedings of the 1990 IEEE Conference on Software Maintenance*, IEEE Computer Society Press, 1990.

LIST OF REFERENCES

1. Kitfield, James, "Is Software DOD's Achilles' Heel?" *Military Forum,* July 1989.
2. United States General Accounting Office, *Mission-Critical Systems: Defense Attempting to Address Major Software Challenges,* December 1992.
3. Defense Systems Management College, *Mission Critical Computer Resources Management Guide,* 1990.
4. United States General Accounting Office, *Embedded Computer Systems: Defense Does Not Know How Much It Spends on Software,* July 1992.
5. Jones, Capers, *Assessment and Control of Software Risks,* Prentice-Hall, 1994.
6. Department of the Army, *Special Operations Aircraft Business Plan,* March 1994.
7. Department of the Army, *Computer Resources Management Plan for the MH-60K and MH-47E Special Operations Aircraft Integrated Avionics Subsystem,* 15 June 1992.
8. Department of the Army, *Special Operations Aircraft Post Deployment Software Support Economic Analysis,* 25 August 1993.
9. Glass, Robert, L., "The "Software Crisis" - Is It a Matter of "Guts Management"?" *Software Management,* Donald J. Reifer, editor, IEEE Computer Society Press, 1993.
10. Department of Defense, *Report of the Defense Science Board Task Force on Military Software,* September 1987.
11. Marciniak, John, J. and Reifer, Donald, J., "Software Acquisition Management," *Software Management,* Donald J. Reifer, editor, IEEE Computer Society Press, 1993.
12. Department of Defense Directive 5000.1, *Defense Acquisition,* February 23, 1991.
13. Department of Defense Instruction 5000.2, *Defense Acquisition Management Policies and Procedures,* February 23, 1991.
14. Secretary of Defense Policy Memorandum, *Specifications & Standards - A New Way of Doing Business,* 29 June 1994.
15. Department of Defense Standard, DOD-STD-2167A, *Defense System Software Development,* 29 February 1988.
16. Department of Defense Standard, DOD-STD-2168, *Defense System Software Quality Program,* 29 April 1988.
17. Department of Defense, MIL-STD-498, *Software Development and Documentation,* 5 December 1994.

18. Thayer, Richard, H., "Software Engineering Project Management: A Top-Down View," *Tutorial: Software Engineering Project Management*, Richard H. Thayer, editor, IEEE Computer Society Press, 1987.

19. Marciniak, John, J., and Reifer, Donald, J., *Software Acquisition Management*, John Wiley & Sons, 1990.

20. Department of the Air Force, Software Technology Support Center, *Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapons Systems, Command and Control Systems, Management Information Systems,* Version 1.1, February 1995.

21. Grady, Robert, B., *Practical Software Metrics for Project Management and Process Improvement,* Prentice-Hall, 1992.

22. Humphrey, Watts, S., "Software Process Improvement at Hughes Aircraft," *Software Management,* Donald J. Reifer, editor, IEEE Computer Society Press, 1993.

23. Arthur, Lowell, Jay, *Software Evolution: The Software Maintenance Challenge*, John Wiley & Sons, 1988.

24. Piersall, James, "The Importance of Software Support to Army Readiness," *Army Research, Development and Acquisition Bulletin*, January-February 1994.

25. Department of Defense, MIL-HDBK-347, *Military Handbook: Mission-Critical Computer Resources Software Support*, 22 May 1990.

26. Martin, James, and McClure, Carma, *Software Maintenance: The Problem and Its Solutions*, Prentice-Hall, 1983.

27. Osborne, Wilma, M., "Building and Sustaining Software Maintainability," *Proceedings of the 1987 IEEE Conference on Software Maintenance*, IEEE Computer Society Press, 1987.

28. Department of the Army Letter 70-95-XX, *Army Post Deployment Software Support (PDSS) for Mission Critical Computer Resources (MCCR) Systems*, 24 April 1995.

29. Vollman, Thomas, "Transitioning From Development to Maintenance," *Proceedings of the 1990 IEEE Conference on Software Maintenance*, IEEE Computer Society Press, 1990.

30. Institute of Electrical and Electronics Engineers, IEEE Std 1219-1993, *IEEE Standard for Software Maintenance*, June 2, 1993.

31. Pigoski, Thomas, M., *Life Cycle Strategy: Software Support on the Front Line*, Software Maintenance News, Inc., 1995.

32. Department of the Army, Aviation Systems Command, *Acquisition Plan for the Special Operations Forces Helicopter Modification Project (SOF MOD)*, 3 April 1987.

33. United States General Accounting Office, *Special Operations Forces: Army Plans Highly Concurrent Acquisition Strategy for Costly Helicopters,* September 1990.

34. Department of the Army, Secretary of the Army Letter to The Honorable Sam Nunn, 13 June 1990.

35. Department of Defense, "Special Operations Forces: Army Plans Highly Concurrent Acquisition Strategy for Costly Helicopters," *Response to the GAO Final Report Findings and Recommendations.*

36. Hyde, James, C., "Army Still Wrestling with Software Glitches in Special Ops Helo Programs," *Armed Forces Journal International,* April 1993.

37. LORAL Federal Systems Company, *Army SOA IAP Metrics,* March 3, 1995.

38. Department of the Army, Special Operations Aircraft Product Management Office, *SOA Software Briefing Pack,* 8 October 1993.

39. Site Visit and Briefing at LORAL Federal Systems Company, Owego, N.Y., January 12, 1995.

40. LORAL Federal Systems Company, *Software Development Plan for the U.S. Army Special Operations Aircraft Post Deployment Software Support Contract,* October 24, 1994.

41. Sinnott, John, et al., *The MH-60K: A Special Rotorcraft for Special Operations,* United Technologies, Sikorsky Aircraft, Stratford, CT., unpublished paper.

42. Interview Granted, Rivera, Jorge, former SOA software engineer, U.S. Army Communications-Electronics Command, 11 October 1995.

43. Site Visit and Briefing at the Software Engineering Directorate, U.S. Army Missile Command, Huntsville, Alabama, October 27, 1994.

44. Site Visit and Briefing at the Software Development and Life Cycle Support Division, Naval Air Warfare Center, Warminster, Pennsylvania, November, 18, 1994.

45. Department of the Army, Aviation and Troop Command, *Minutes of the Special Operations Aircraft PDSS Process Action Team Meeting,* 19-20 May 93.

46. Correspondence with Dr. Vernon Allen, Director of Life Cycle Software Engineering Directorate, U.S. Army Aviation and Troop Command, September 29, 1995.

47. Department of the Army, Aviation and Troop Command, *SOA PDSS Contract Number DAAJ09-94-C-0386,* 25 July 1994.

48. Department of the Army, Technology Applications Program Office, *Special Operations Organization Briefing Pack.*

49. Department of the Army, Technology Applications Program Office, *SOA PDSS Briefing Pack.*

50. Interview Granted, Hopkins, Gerald, J., Major, Project Officer, Technology Applications Program Office, 22 March 1995.

51. Interview Granted, Hampp, Ralph, SOA Program Manager, LORAL Federal Systems Company, 26 September 1995.

52. Department of the Army, Technology Applications Program Office, *Statement of Work for the U.S. Army Special Operations Aircraft (SOA) Post Deployment Software Support*, Revision A, Feb 13, 1995.

53. Department of the Army, Technology Applications Program Office, *Statement of Work Update for the U.S. Army Special Operations Aircraft (SOA) Year Two of Post Deployment Software Support*, Revision A, Aug 21, 1995.

54. Interview Granted, Walker, Robert, Chief Warrant Officer, Project Officer, Systems Integration Management Office, 15 November 1995.

55. Interview Granted, Hirsh, Vicki, Contracting Officer, U.S. Army Aviation and Troop Command, 23 March 1995.

56. Interview Granted, Hope, Michael, Software Engineer, U.S. Army Aviation and Troop Command, 13 October 1995.

57. Defense Systems Management College, *Systems Engineering Management Guide*, January 1990.

58. Researcher Notes, PDSS Negotiations at the U.S. Army Aviation and Troop Command, August 30, 1995.

59. Defense Contract Audit Agency, Audit Report Number 6351-95A21000008, 30 December 1994.

60. Correspondence with Schrank, Adam, Software Engineer, Naval Air Systems Command, October 4, 1995.

61. LORAL Federal Systems Company, *PDSS Cost Data*, October 30, 1995.

62. Department of the Army, Product Management Office, *Special Operations Aircraft, Release 12.0 SLOC Update*, March 9, 1995.

63. Haimes, Yacov, Y., and Chittister, Clyde, "An Acquisition Process for the Management of Nontechnical Risks Associated with Software Development," *Acquisition Review Quarterly*, Spring 1995.

64. United States General Services Administration, *Federal Acquisition Regulation*, December 1993.

65. Department of the Army, PATRIOT Program Office Memo, *Subject: Maintenance of Army Software*, 8 January 1991.

66. Mintzberg, Henry and Quinn, James Brian, *The Strategy Process*, Prentice Hall, 1992.

67. Naval Postgraduate School, *Financial Management in the Armed Forces: Companion Guide*, MN 3154 class handout, Fall 1994.

68. Templin, Carl, R., "Defense Contracting Buyer-Seller Relationships: Theoretical Approaches," *Acquisition Review Quarterly*, Spring 1994.

69. Landeros, Robert and Monczka, Robert, M., "Cooperative Buyer/Seller Relationships and a Firm's Competitive Posture," *Journal of Purchasing and Materials Management*, Fall 1989.

70. United States General Services Administration, *Federal Acquisition Circular 90-29; FAR Case 95-10, Item 1, FAR Guiding Principles*, August 3, 1995.

71. Attanasio, Henry, *Contracting for Computer Software within the Department of the Navy*, Thesis, Naval Postgraduate School, June 1990.

72. Department of Defense, *Defense Federal Acquisition Regulation Supplement*, 1991.

73. National Aeronautics and Space Administration, *Award Fee Contracting Guide*, June 1994.

74. Hunter, Mark, T., *Award Fee in Software Acquisition*, Thesis, Air Force Institute of Technology, September 1992.

75. Interview Granted, Rennick, Claire, E., Deputy Product Manager, Advanced Field Artillery Tactical Data System, 15 February 1995.

76. Interview Granted, Ridgway, Errol, L., Contracting Officer, NASA Ames Research Center, 17 February 1995.

77. Garrett, Gregory, A., "Contracting with an Award Fee-It Works! (But Nobody Said It Would Be Easy)," *Program Manager*, May-June 1989.

78. Department of the Navy, *Software Metrics Program Handbook AVDIV-HDBK-7*, 24 September 1994.

79. Guenther, Otto, J., Lieutenant General, "An Army Perspective on Software Development," *Cross Talk*, May 1995.

80. Sorensen, Reed, "A Comparison of Software Development Methodologies," *Cross Talk*, January 1995.

81. Butler, Kelly, L., "The Economic Benefits of Software Process Improvement," *Cross Talk*, July 1995.

82. United States House of Representatives, 103D Congress, 2d Session, *Conference Report on the Federal Acquisition Streamlining Act of 1994*, August 21, 1994.

83. Basili, Victor, R., and Rombach, H., Dieter, "Tailoring the Software Process to Project Goals and Environments," *Proceedings of the 1987 IEEE International Conference on Software Engineering, IEEE Computer Society Press*, 1987.

84. Grady, Robert, B., "Measuring and Managing Software Maintenance," *IEEE Software*, September 1987.

LIST OF INTERVIEWS

1. Allen, Vernon, Director of Life-Cycle Software Engineering, U.S. Army Aviation and Troop Command, March 1995.
2. Buckner, Randy, Technical Division Chief, Special Operations Aircraft Product Manager's Office, March 1995.
3. Blackwell, Frank, Software Engineer, U.S. Army Missile Command, August 1995.
4. Hampp, Ralph, SOA Program Manager, LORAL Federal Systems Company, September 1995.
5. Hanrahan, Robert, U.S. Air Force Software Technology Support Center, August 1995.
6. Hirsh, Vicki, Contracting Officer, U.S. Army Aviation and Troop Command, March 1995.
7. Hope, Michael, Software Engineer, U.S. Army Aviation and Troop Command, October 1995.
8. Hopkins, Gerald, J., Major, Project Officer, Technology Applications Program Office, March 1995.
9. Hucke, Kay, Contracting Officer, Technology Applications Program Office, March 1995.
10. Kennedy, Kerry, Software Engineer, U.S. Army Missile Command, August 1995.
11. Pigoski, Thomas, M., President, Technical Software Services, Inc., September 1995.
12. Rennick, Claire, E., Deputy Product Manager, Advanced Field Artillery Tactical Data System, February 1995.
13. Ridgway, Errol, L., Contracting Officer, NASA Ames Research Center, February 1995.
14. Rivera, Jorge, former SOA software engineer, U.S. Army Communications-Electronics Command, October 1995.
15. Rogers, Michael, W., Lieutenant Colonel, Product Manager, Special Operations Aircraft, March 1995.
16. Schrank, Adam, Software Engineer, Naval Air Systems Command, October 1995.
17. Walker, Robert, Chief Warrant Officer, Project Officer, Systems Integration Management Office, November 1995.

BIBLIOGRAPHY

Allen, Vernon, Director of Life Cycle Software Engineering Directorate, U.S. Army Aviation and Troop Command, electronic mail correspondence, September 29, 1995.

Adams, Charles, J. et al, *NDI Acquisition: An Alternative to "Business as Usual," Report of the Defense Systems Management College 1991-1992 Military Research Fellows*, 1992.

Arthur, Lowell, Jay, *Software Evolution: The Software Maintenance Challenge*, John Wiley & Sons, 1988.

Attanasio, Henry, *Contracting for Computer Software within the Department of the Navy*, Thesis, Naval Postgraduate School, June 1990.

Basili, Victor, R., and Rombach, H., Dieter, "Tailoring the Software Process to Project Goals and Environments," *Proceedings of the 1987 IEEE International Conference on Software Engineering, IEEE Computer Society Press*, 1987.

Butler, Kelly, L., "The Economic Benefits of Software Process Improvement," *Cross Talk*, July 1995.

Butts, Forrest, F. and Johndro, Anthony, C., *Guidelines for Ensuring Software Supportability in Systems Developed Under the Integrated Weapon System Management Concept*, Thesis, Air Force Institute of Technology, 1993.

Defense Systems Management College, *Mission Critical Computer Resources Management Guide*, 1990.

Defense Systems Management College, *Systems Engineering Management Guide*, January 1990.

Defense Contract Audit Agency, Audit Report Number 6351-95A21000008, 30 December 1994.

Department of Defense, *Defense Federal Acquisition Regulation Supplement*, 1991.

Department of Defense Directive 5000.1, *Defense Acquisition*, February 23, 1991.

Department of Defense Instruction 5000.2, *Defense Acquisition Management Policies and Procedures*, February 23, 1991.

Department of Defense, MIL-HDBK-347, *Military Handbook: Mission-Critical Computer Resources Software Support*, 22 May 1990.

Department of Defense, MIL-STD-498, *Software Development and Documentation*, 5 December 1994.

Department of Defense Standard, DOD-STD-2167A, *Defense System Software Development*, 29 February 1988.

Department of Defense, DOD-STD-2168, *Defense System Software Quality Program,* 29 April 1988.

Department of Defense, "Special Operations Forces: Army Plans Highly Concurrent Acquisition Strategy for Costly Helicopters," *Response to the GAO Final Report Findings and Recommendations.*

Department of Defense, *Report of the Defense Science Board Task Force on Military Software,* September 1987.

Department of the Air Force, Software Technology Support Center, *Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapons Systems, Command and Control Systems, Management Information Systems,* Version 1.1, February 1995.

Department of the Army, Aviation Systems Command, *Acquisition Plan for the Special Operations Forces Helicopter Modification Project (SOF MOD),* 3 April 1987.

Department of the Army, PATRIOT Program Office Memo, *Subject: Maintenance of Army Software,* 8 January 1991.

Department of the Army, Secretary of the Army Letter to The Honorable Sam Nunn, 13 June 1990.

Department of the Army, *Special Operations Aircraft Post Deployment Software Support Economic Analysis,* 25 August 1993.

Department of the Army, Special Operations Aircraft Product Management Office, *SOA Software Briefing Pack,* 8 October 1993.

Department of the Army, Aviation and Troop Command, *Minutes of the Special Operations Aircraft PDSS Process Action Team Meeting,* 19-20 May 93.

Department of the Army, Aviation and Troop Command, *SOA PDSS Contract Number DAAJ09-94-C-0386,* 25 July 1994.

Department of the Army, *Special Operations Aircraft Business Plan,* March 1994.

Department of the Army, Technology Applications Program Office, *Special Operations Organization Briefing Pack.*

Department of the Army, Technology Applications Program Office, *SOA PDSS Briefing Pack.*

Department of the Army, Technology Applications Program Office, *Statement of Work for the U.S. Army Special Operations Aircraft (SOA) Post Deployment Software Support,* Revision A, Feb 13, 1995.

Department of the Army, Technology Applications Program Office, *Statement of Work Update for the U.S. Army Special Operations Aircraft (SOA) Year Two of Post Deployment Software Support*, Revision A, Aug 21, 1995.

Department of the Navy, *Software Metrics Program Handbook AVDIV-HDBK-7*, 24 September 1994.

United States Army Audit Agency, *Report of Audit of the Special Operations Aircraft*, 24 April 1992.

Department of the Army Letter 70-95-XX, *Army Post Deployment Software Support (PDSS) for Mission Critical Computer Resources (MCCR) Systems*, 24 April 1995.

Department of the Army, Communications-Electronics Command, *Computer Resources Management Plan for the MH-60K and MH-47E Special Operations Aircraft Integrated Avionics Subsystem,* 15 June 1992.

Department of the Army, Product Management Office, *Special Operations Aircraft, Release 12.0 SLOC Update*, March 9, 1995.

Department of the Army, U.S. Army Missile Command, *Methodology for the Management of Software Acquisition*, June 1991.

Francom, Gerald, Lee, *A Proposal to Change the Federal Acquisition Regulation: Recognizing the Award Fee Incentive in Fixed-Price Contracts*, Thesis, Naval Postgraduate School, June 1989.

Garrett, Gregory, A., "Contracting with an Award Fee-It Works! (But Nobody Said It Would Be Easy)," *Program Manager*, May-June 1989.

Glass, Robert, L., "The "Software Crisis" - Is It a Matter of "Guts Management"?" *Software Management,* Donald J. Reifer, editor, IEEE Computer Society Press, 1993.

Grady, Robert, B., *Practical Software Metrics for Project Management and Process Improvement,* Prentice-Hall, 1992.

Grady, Robert, B., "Measuring and Managing Software Maintenance," *IEEE Software*, September 1987.

Guenther, Otto, J., Lieutenant General "An Army Perspective on Software Development," *Cross Talk*, May 1995.

Haimes, Yacov, Y., and Chittister, Clyde, "An Acquisition Process for the Management of Nontechnical Risks Associated with Software Development," *Acquisition Review Quarterly*, Spring 1995.

Humphrey, Watts, S., *Managing the Software Process*, Addison-Wesley, 1989.

Humphrey, Watts, S., "Software Process Improvement at Hughes Aircraft," *Software Management,* Donald J. Reifer, editor, IEEE Computer Society Press, 1993.

Hunter, Mark, T., *Award Fee in Software Acquisition*, Thesis, Air Force Institute of Technology, September 1992.

Hyde, James, C., "Army Still Wrestling with Software Glitches in Special Ops Helo Programs," *Armed Forces Journal International*, April 1993.

Institute of Electrical and Electronics Engineers, IEEE Std 1219-1993, *IEEE Standard for Software Maintenance*, June 2, 1993.

Jones, Capers, *Assessment and Control of Software Risks*, Prentice-Hall, 1994.

Kind, Peter, A., Lieutenant General, *Software Initiatives for System Acquisition*, briefing pack, 15 June 1993.

Kitfield, James, "Is Software DOD's Achilles' Heel?" *Military Forum*, July 1989.

Landeros, Robert and Monczka, Robert, M., "Cooperative Buyer/Seller Relationships and a Firm's Competitive Posture," *Journal of Purchasing and Materials Management*, Fall 1989.

Lientz, B.P. and Swanson, E.B., *Software Maintenance Management*, Addison-Wesley, 1980.

LORAL Federal Systems Company, *Army SOA IAP Metrics*, March 3, 1995.

LORAL Federal Systems Company, *PDSS Cost Data*, October 30, 1995.

LORAL Federal Systems Company, *Software Development Plan for the U.S. Army Special Operations Aircraft Post Deployment Software Support Contract*, October 24, 1994.

Marciniak, John, J. and Reifer, Donald, J., "Software Acquisition Management," *Software Management*, Donald J. Reifer, editor, IEEE Computer Society Press, 1993.

Marciniak, John, J., and Reifer, Donald, J., *Software Acquisition Management*, John Wiley & Sons, 1990.

Martin, James, and McClure, Carma, *Software Maintenance: The Problem and Its Solutions*, Prentice-Hall, 1983.

Mintzberg, Henry and Quinn, James Brian, *The Strategy Process*, Prentice Hall, 1992.

National Aeronautics and Space Administration, *Award Fee Contracting Guide*, June 1994.

Naval Postgraduate School, *Financial Management in the Armed Forces: Companion Guide*, MN 3154 class handout, Fall 1994.

Osborne, Wilma, M., "Building and Sustaining Software Maintainability," *Proceedings of the 1987 IEEE Conference on Software Maintenance*, IEEE Computer Society Press, 1987.

Piersall, James, "The Importance of Software Support to Army Readiness," *Army Research, Development and Acquisition Bulletin*, January-February 1994.

Pigoski, Thomas, M., *Life Cycle Strategy: Software Support on the Front Line*, Software Maintenance News, Inc., 1995.

Researcher Notes, PDSS Negotiations at the U.S. Army Aviation and Troop Command, August 30, 1995.

Schade, Don, F., *Fixed-Price-Award-Fee: An Economic, Motivational, and Contracting Theory Analysis*, Thesis, Naval Postgraduate School, December 1990.

Schrank, Adam, Software Engineer, Naval Air Systems Command, electronic mail correspondence, October 4, 1995.

Secretary of Defense Policy Memorandum, *Specifications & Standards - A New Way of Doing Business*, 29 June 1994.

Sinnott, John, et al., *The MH-60K: A Special Rotorcraft for Special Operations*, United Technologies, Sikorsky Aircraft, Stratford, CT., unpublished paper.

Site Visit and Briefing at LORAL Federal Systems Company, Owego, N.Y., January 12, 1995.

Site Visit and Briefing at the Software Development and Life Cycle Support Division, Naval Air Warfare Center, Warminster, Pennsylvania, November, 18, 1994.

Site Visit and Briefing at the Software Engineering Directorate, U.S. Army Missile Command, Huntsville, Alabama, October 27, 1994.

Sorensen, Reed, "A Comparison of Software Development Methodologies," *Cross Talk*, January 1995.

Templin, Carl, R., "Defense Contracting Buyer-Seller Relationships: Theoretical Approaches," *Acquisition Review Quarterly*, Spring 1994.

Thayer, Richard, H., "Software Engineering Project Management: A Top-Down View," *Tutorial: Software Engineering Project Management*, Richard H. Thayer, editor, IEEE Computer Society Press, 1987.

United States General Accounting Office, *Embedded Computer Systems: Defense Does Not Know How Much It Spends on Software*, July 1992.

United States General Accounting Office, *Mission-Critical Systems: Defense Attempting to Address Major Software Challenges*, December 1992.

United States General Accounting Office, *Special Operations Forces: Army Plans Highly Concurrent Acquisition Strategy for Costly Helicopters*, September 1990.

United States General Services Administration, *Federal Acquisition Circular 90-29; FAR Case 95-10, Item 1, FAR Guiding Principles*, August 3, 1995.

United States General Services Administration, *Federal Acquisition Regulation*, December 1993.

United States House of Representatives, 103D Congress, 2d Session, *Conference Report on the Federal Acquisition Streamlining Act of 1994*, August 21, 1994.

Vollman, Thomas, "Transitioning From Development to Maintenance," *Proceedings of the 1990 IEEE Conference on Software Maintenance*, IEEE Computer Society Press, 1990.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 Kingman Rd., STE 0944
   Fort Belvoir, VA 22060-6218

2. Library, Code 13 2
   Naval Postgraduate School
   Monterey, CA 93943-5101

3. Defense Logistics Studies Information Exchange 1
   U.S. Army Logistics Management Center
   Fort Lee, VA 23801-6043

4. Acquisition Library 1
   Department of Systems Management
   Naval Postgraduate School
   Monterey, CA 93943-5103

5. OASA (RDA) 1
   ATTN: SARD-ZAC
   103 Army Pentagon
   Washington, DC 20310

6. ODISC4 1
   ATTN: SAIS-ADW (Mr. Robert Schwenk)
   107 Army Pentagon
   Washington, DC 20310-0107

7. U.S. Army Aviation and Troop Command 2
   ATTN: AMSAT-D-T (MAJ Hopkins)
   4300 Goodfellow Blvd.
   St. Louis, MO 63120-1798

8. U.S. Army Aviation and Troop Command 1
   ATTN: AMSAT-A-PMCA (Vicki Hirsh)
   4300 Goodfellow Blvd.
   St. Louis, MO 63120-1798

9. U.S. Army Aviation and Troop Command 2
ATTN: AMSAT-A-PMBB (Archie Ringgenberg)
4300 Goodfellow Blvd.
St. Louis, MO 63120-1798

10. LORAL Federal Systems Company 1
ATTN: Mr. Ralph Hampp
1801 State Route 17C
Owego, NY 13827-3998

11. Thomas M. Pigoski 1
Technical Software Services, Inc.
31 West Garden Street, Suite 100
Pensacola, FL 32501-5615

12. Prof. David V. Lamm (Code SM/Lt) 5
Naval Postgraduate School
Monterey, CA 93943-5103

13. Prof. Martin J. McCaffrey (Code SM/Mf) 5
Naval Postgraduate School
Monterey, CA 93943-5103

14. Prof. W. Max Woods (Code OR/ Wo) 1
Naval Postgraduate School
Monterey, CA 93943-5219

15. LTC John Dillard (Code SM/Dj) 1
Naval Postgraduate School
Monterey, CA 93943-5103

16. Scott C. Dolloff 2
PSC Box 3500
MacDill Air Force Base, FL 33621